A METHOD TO PERFORM FREE-VIBRATION ANALYSIS OF STRUCTURES

BY

HIRA LAL KAUL

DEPARTMENT OF CIVIL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

AUGUST 1971

A METHOD TO PERFORM FREE-VIBRATION ANALYSIS OF STRUCTURES

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

BY

HIRA LAL KAUL

to the

DEPARTMENT OF CIVIL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

AUGUST 1971

TO

MY PARENTS

CERTIFICATE

This is to certify that the thesis entitledd 'A Method to PERFORM FREE VIBRATION ANALYSIS OF STRUCTURES' by Hira Lal Kaul is a record of work carried out under my supervision and has not been submitted elsewhere for a degree.

(M.P. KAPOOR)
ASSISTANT PROFESSOR
DEPARTMENT OF CIVIL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

ACKNOWLEDGEMENTS

The author expresses his appreciation and gratitude to:

His advisor, Dr. Mahesh P. Kapoor, whose encouragement and continuous guidance were vital to the success of this effort.

To all my friends for their constructive help and criticism.

The staff of IIT Kanpur Computer Centre for providing facilities on IBM 7044 Digital Computer.

Mr. S.N. Pandey for the excellent typing of the manuscript and, also, for his patience which is sincerely appreciated.

CONTENTS

SYNOPSIS

of the

Dissertation on

A METHOD TO PERFORM FREE VIBRATION ANALYSIS OF STRUCTURES

A THESIS SUBMITTED
in partial fulfilment of the requirements
for the Degree of
MASTER OF TECHNOLOGY

by

Hira Lal Kaul

to the

DEPARTMENT OF CIVIL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
August 1971

The mode shapes, associated natural frequencies and nature and amount of damping govern the dynamic response of structures. This work presents a minimization method of extracting the eigenvalues and eigenvectors of a structural system having finite degrees of freedom. The structures considered are supposed to be linear and undamped. The Rayleigh quotient has been minimized successively in reduced subspaces to obtain the second and higher frequencies and the mode shapes associated with them. Conjugate Gradient Method of minimization has been used. Its modest demands on Computer storage space have been further supplemented

by performing the analysis which does not require the assembly of structural stiffness and mass matrices. The method seems to be particularly useful for large sized problems where only first few eigenvalues and eigenvectors are desired.

The study of free vibrations is usually an intermediate step to perform the forced vibration analysis. The method has been applied to simple structures like cantilever beam, portal frame and skew bridge and results obtained compare very well.

CHAPTER - I

INTRODUCTION

Performing a discrete element analysis of the structure, the matrix formulation of the general dynamic response problem for the undamped structure is:

$$[M]\,\ddot{\vec{Y}} + [K]\,\vec{Y} = \vec{F}\,(t) \qquad \ldots (1)$$

where the n x n matrices $[M]$ and $[K]$ are respectively the mass matrix and the stiffness matrix of the structure. These matrices are real and symmetric and their order n corresponds to the degrees of freedom to which an infinite degree of freedom continuous structure has been discretized. The n x 1 vectors $\ddot{\vec{Y}}$, $\vec{Y}$ and $\vec{F}\,(t)$ represent the absolute acceleration, absolute displacement and load respectively. The set of equations Eqn. (1), consist of a set of coupled second order ordinary differential equations. These are usually solved by uncoupling the equations through matrix transformation. The uncoupled equations resemble the equation of motion of a single degree of freedom system. The dynamic response is finally obtained by the linear superposition of the solution of these uncoupled equations. The matrix which performs uncoupling of the set of Eqn.(1)

requires the free vibration analysis of the structure which in turn means the solution to the set of equations:

$$[M]\ \ddot{\vec{Y}} + [K]\ \vec{Y} = 0 \qquad \dots (2)$$

The above set of equations are reduced to an eigenvalue problem

$$[K]\ \vec{X} = \lambda [M]\ \vec{X} \qquad \dots (3)$$

where λ is the scalar quantity called eigenvalue which represents the square of the natural frequency of the system and $\vec{X}$ is the $(n \times 1)$ eigenvector called the principal mode shape associated with the particular value λ.

This work presents a method to obtain the eigenvalues λ and associated eigenvectors $\vec{X}$. The method presented here is iterative and exploits the minimization technique and is expected to be specifically useful for determination of the first few eigenvalues and associated eigenvectors for large sized structures i.e. where n is large.

1.1 METHODS OF EIGENSOLUTION:

By eigensolution is meant the solution of the eigenproblem, Eqn. (3), which in turn means the determination of the scalar quantities λ and their associated

modeshapes $\vec{X}$.

Various methods of eigensolution known can be categorized as:

(a) Transformation Methods

(b) Iterative Methods.

Transformation methods require to convert the general eigenvalue problem:

$$[K]\vec{X} = [M]\vec{X} \quad \text{into a special form}$$

$$[A]\vec{X} = \lambda\vec{X}$$

and then involve a series of similarity transformations which operate on the matrix $[A]$. This requires the storage of the full n x n matrix $[A]$ in the computer. If n is large, it occupies sufficient computer space. Furthermore, all transformation methods like Givens, Householder, Jacobi etc., have been found to be efficient for the complete eigensolution i.e., determination of all the n eigenvalues and associated eigenvectors of the set of equations Eqn. (3), presuming that there exist n distinct eigenvalues.

However, a reasonably accurate dynamic response analysis of structures can be performed with the knowledge of only the first few eigenvalues and eigenvectors from the lowest end of the spectrum and hence the complete eigensolution is not required. Therefore, transformation methods are not very efficient for use in determining the dynamic response of large sized structures.

Iterative methods like Power Method, also require the initial conversion of general eigenvalue problem to a particular one and thus suffer with the same disadvantage as the transformation methods so far as computer space is concerned. Furthermore, these methods are not very stable and require periodic cleaning inorder to obtain eigenvalues higher than the fundamental one.

The iterative method described and used in the present work has the advantage that it does not require any initial conversion of general eigenvalue problem to a particular form. Moreover, the method does not require the assembly of (n x n) structural stiffness and mass matrices. The entire algorithm uses the stiffness and mass matrices of the elements which compose the structure and their order is very small compared to n.

Lastly the method has been observed to be numerically stable and efficient for the determination

of partial eigensolution i.e., the first few natural frequencies and mode shapes of the vibrating structure.

Chapter 2 describes the formulation of the method for eigensolution. Chapter 3 describes the algorithm used to obtain the eigensolution. Chapter 4 gives the results for the class of structures, to which the method has been applied.

CHAPTER - II

FORMULATION OF THE METHOD OF EIGENSOLUTION*

An iterative method using the well known property of Rayleigh quotient is presented which can be applied directly to the eigenproblem:

$$[K]\vec{X} = \lambda [M]\vec{X}$$

It uses the property of the Rayleigh quotient:

$$R(\vec{X}) = \frac{\vec{X}^T [K] \vec{X}}{\vec{X}^T [M] \vec{X}},$$ that it equals the eigenvalue when the eigenvector is substituted into it and that it is stationary in the neighbourhood of an eigenvector. The Rayleigh quotient is minimized to obtain the lowest eigenvalue and the associated eigenvector. The minimization is done numerically using the conjugate gradient method. The intermediate eigenvalues and

---

* The method was developed and is described in detail in reference no. For completeness of presentation, it is briefly described here.

the associated eigenvectors are obtained by using a gradient projection scheme for constraining the minimization search to the subspace M - orthogonal to the previously determined eigenvectors.

The advantage of the formulation

$$R(\vec{X}) = \frac{\vec{X}^T [K] \vec{X}}{\vec{X} [M] \vec{X}},$$

is that both the numerator and the denominator, as well as all of the other quantities needed for iteration procedure for all of the eigenvalue desired, can be computed without having the assembled $[K]$ and $[M]$ matrices at hand. Since, the numeration is twice the strain energy for a give $\vec{X}$ (the displacement vector), and the denominator is twice the maximum kinetic energy of the structure, the summation of potential and kinetic energies of the individual elements of the discretized structural model gives us the numerator and the denominator.

2.1 FORMULATION OF THE PROBLEM:

The eigenproblem can be written as:

$$\left\{ [K] - \lambda [M] \right\} \vec{X} = 0$$

If $\vec{X}$ is its solution, then b $\vec{X}$ is also a solution for any non-zero value of the scalar b, and thus

the eigenvector corresponding to any eigenvalue is arbitrary to the extent of a scalar multiplier. The Rayleigh quotient is not defined at the origin and thus care has to be taken while minimizing. The redundant degrees of freedom which prevents the determination of absolute magnitude of the components of the eigenvector, can be eliminated by an arbitrary normalization and the simplest normalization is to set any non-zero component of the eigenvector equal to one. The value of the Rayleigh quotient is bounded by the lowest and highest eigenvalues of the physical system.

Thus, the minimization of the Rayleigh quotient will yield the lowest eigenvalue. The minimization problem to yield the lowest eigenvalue can thus be stated as:

Find $\vec{X} = \vec{X}_1$ such that

$$R(\vec{X}_1) \equiv \frac{\vec{X}_1^T [K] \vec{X}_1}{\vec{X}_1 [M] \vec{X}_1},$$

is minimum, subject to, $x_{q1} \equiv \vec{X}_1^T \vec{e}_q = 1$ where x_{q1} is the normalizing or reference component and $\vec{e}_q$ is a vector with its qth component as one and zero elsewhere (i.e., $\vec{e}_q$ is a unit co-ordinate vector for the qth co-ordinate).

Once the lowest eigenvalue $\lambda_1 \equiv R(\vec{X}_1)$ is known, the next higher or (second) eigenvalue and its associated eigenvector can be determined by posing a new minimization problem:

Find $\vec{X} = \vec{X}_2$, such that

$$R(\vec{X}_2) = \frac{\vec{X}_2^T \left[K\right] \vec{X}_2}{\vec{X}_2^T \left[M\right] \vec{X}_2}$$

is minimum, subject to

$$\vec{X}_2^T \vec{e}_r = 1$$

and $\vec{X}_2^T \left[M\right] \vec{X}_1 = 0.$

In the subspace defined by the above constraints, the Rayleigh quotient takes on a unique minimum at the eigenvector associated with the second lowest eigenvalue. The second constraint represents the imposition of the M - orthogonality condition between $\vec{X}_1$ and $\vec{X}_2$. Geometrically speaking, these constraints merely restrict the position of vector space in which the search for the second eigenvector is carried out and in this restricted subspace, R has a minimum corresponding to λ_2.

Determination of the third and subsequent eigenvalues and the associated eigenvectors is accomplished by solving

a sequence of problems similar, as for the determination of 2nd eigenvalue. The only change is that each time one additional equation of constraint has to be imposed on the minimization problem to satisfy the condition that the eigenvector currently being sought is M - orthogonal to all of the previously determined eigenvectors. The problem of determining the lth eigenvalue ($2 \leq l \leq n$) can thus be written

Find $\vec{X} = \vec{X}_l$, such that

$$R(\vec{X}_l) = \frac{\vec{X}_l^T [K] \vec{X}_l}{\vec{X}_l^T [M] \vec{X}_l}$$

Subject to $\vec{X}_l^T \vec{e}_j = 1$

and $\vec{X}_l^T [M] \vec{X}_i = 0$, $\quad i = 1,2 \, . \, . \, . \, , l - 1$

where $\vec{X}_i$, $i = 1,2, \ldots, l - 1$ are assumed to be known when the lth eigenvector is being sought.

Denoting $[M]\vec{X}_i \equiv \vec{V}_i$, the above constraint equation can be written as

$$\vec{X}_l^T \vec{V}_i = 0, \quad i = 1,2, \ldots, l - 1.$$

CHAPTER - III

MINIMIZATION - ALGORITHM

The gradient methods are usually more powerful to minimize a function of n variables as they use the local information about the rate of change of function with respect to the changes in the variables. Hence one of the gradient methods is being used here for minimization of the Rayleigh quotient. The Rayleigh quotient as a function of the n variables $(X_1, X_2, \ldots, X_n) \equiv \vec{X}$ is differentiable, and its gradient vector

$$\nabla R \equiv \vec{g} = \frac{2[K]\vec{X}}{\vec{X}^T[M]\vec{X}} - \frac{(\vec{X}[K]\vec{X})}{(\vec{X}^T[M]\vec{X})^2} \cdot 2[M]\vec{X}$$

$$= \frac{2([K]\vec{X} - R[M]\vec{X})}{(\vec{X}^T[M]\vec{X})}$$

is easily computed.

The steepest descent method which is often faced with the convergence difficulties is modified for rapid convergence by conjugate gradient method. The Flatcher Reeves Conjugate gradient method has been used here, for reasons of its simplicity and its modest storage demands.

The algorithm is as follows:

3.1 FLETCHER AND REEVES METHOD:

Conjugate Gradients:

The convergence difficulties of the steepest descent (gradient) method can be greatly reduced by a very simple modification which converts it to the Conjugate Gradient method.

Here, the entire algorithm reduce to

$$\vec{X}_0 = \text{arbitrary}$$

$$\vec{G}_0 = \nabla F (\vec{X}_0)$$

$$\vec{S}_0 = - \vec{G}_0$$

$$\vec{X}_{i+1} = \vec{X}_i + \alpha_i^* \vec{S}_i$$

$$\vec{G}_{i+1} = \nabla F (\vec{X}_{i+1})$$

$$\beta_i = | G_{i+1}|^2 / |\vec{G}_i|^2$$

$$\vec{S}_{i+1} = - \vec{G}_{i+1} + \beta_i \vec{S}_i$$

Clearly from this definition of $\vec{S}_{i+1}$, it is a linear combination of $\vec{G}_{i+1}$ and $\vec{S}_o, \vec{S}_1, \ldots, \vec{S}_i$

and hence, it is a linear combination of $\vec{G}_0$, $\vec{G}_1$, . . . , $\vec{G}_{i+1}$.

The conjugate gradient method is a rapidly convergent technique, suitable for use when the gradient of the function is readily computed. It generally far surpasses the steepest descent method except in unusual cases and with careful application, it is one of the most effective minimization techniques. In programming the method, attention is to be paid to the questions of the accuracy required in the determination of α^*, a strategy for restarting and a proper scaling.

There are more powerful methods, but they are generally impractical in large complicated problems.

The conjugate gradient method is the most efficient minimization technique as the problem size is increased, inspite of its weaker stability.

The problem of minimizing the Rayleigh quotient function to find the lowest eigenvalue is similar to an unconstrained minimization problem and the algorith can be directly applied. The use of conjugate gradient method for finding the intermediate eigenvalue is possible only when the minimization of the Rayleigh quotient is restricted into a subspace of $\vec{X}$ in which the constraints

$$\vec{X}_1^T \vec{e}_j = 1 \quad \text{and} \quad \vec{X}_1 [M] \vec{X}_i = 0, \; i = 1, 2, - 1 - 1$$

is continuously satisfied. Thus, in order to insure that the search is carried out in the desired subspace of $\vec{X}$, it is necessary (1) to start the iteration with a point in that subspace and (2) to project the gradient vector $\vec{g}$, onto that subspace. Such a projection matrix is generated below:

3.2 PROJECTION MATRIX:

Let $[P]$ be a matrix which has the property that for any vector $\vec{W}$, the vector

$$\vec{W}_p = [P] \vec{W}$$

satisfies, $\vec{W}_p^T \vec{Z}_i = 0, \; i = 1, 2, \ldots, q$

where $\vec{Z}_i$, $i = 1, 2, \ldots, q$ are q linearly independent vectors.

The above Eqn. can also be written in matrix form as

$$\underset{(q \times n)}{[M]^T} \; \underset{(n \times 1)}{\vec{W}_p} = 0$$

where $[N]_{nxq} = [\vec{Z}_1, \vec{Z}_2, \ldots, \vec{Z}_q]$

In otherwords, the matrix operator $[P]$ eliminates from the vector $\vec{W}$ the non-orthogonal components, thus giving the vector $\vec{W}_p$ which is orthogonal to the subspace spanned by the vectors $\vec{Z}_i$, $i = 1,2, \ldots, q$. This idea is differently expressed as:

$$\vec{W}_p = \vec{W} - \sum_{i=1}^{q} u_i \vec{Z}_i$$, or in matrix form as

$$\vec{W}_p = \vec{W} - \underset{(nxq)}{[N]} . \underset{(qx1)}{\vec{u}}$$

where the components of vector $\vec{U}$ are u_i, $i = 1,2, \ldots, q$.

Pre-multiplying above Eqn. by $[N]^T$, we obtain from

$[N]^T \vec{W}_p = 0$

$$[N]^T \vec{W}_p = [N]^T \vec{W} - ([N]^T [N]) \vec{U} = 0$$

$$\therefore \quad \vec{U} = ([N]^T [N])^{-1} [N]^T \vec{W}$$

Thus, we obtain

$$\vec{W}_p = \vec{W} - [N] (N^T N)^{-1} N^T \vec{W}$$

$$\vec{W}_b = \left\{ [I] - [N] (N^T N)^{-1} N^T \right\} \vec{W}$$

$(N^T N)$ will be a $(q \times q)$ symmetric matrix, and is nonsingular since $[N]$ is a $(n \times q)$ matrix composed of q linearly independant columns.

Hence, its inverse exists.

The projection matrix, by comparison is obtained as

$$[P] = \left\{ [I] - [N] \left([N]^T [N] \right)^{-1} [N]^T \right\}$$

For determining the lth eigensolution, a projection matrix

$$[P_l] = \left\{ [I] - \underset{(n \times l)}{[N_l]} \left(\underset{(l \times l)}{[N_l]^T [N_l]} \right)^{-1} \underset{(l \times n)}{[N_l]^T} \right\}$$

where $[N_l] = \left[\vec{e}_j, V_1, \vec{V}_2, \ldots, \vec{V}_{l-1} \right]$

will project the gradient vector $\vec{g}$, on to the subspace of constraints already defined. The column vectors of the matrix $[N_l]$ are linearly independent. Thus

$$\vec{g}_p = [P_l] \vec{g}$$

One thing that warrants attention here is that, since we use $\vec{g}_p$ instead of $\vec{g}$, will the conjugate gradient method actually converge? The answer is

affirmative and it can be shown by constructing the Lagrange function for the unconstrained minimization problem.

STEP BY STEP PROCEDURE:

3.3 (a) Choice of a Starting Point:

A the iterative methods should have a good starting point, otherwise time is un-necessarily wasted inside the minimization procedure. A starting vector as a unit vector $\vec{c}_i$ (the jth element of $\vec{e}_i$ is δ_{ij}) lying along the co-ordinate axes, gives an awfully distorted mode shape in thedynamics problem. A set of random vectors proved to be superior contrary to the expectation and in this work they were used as starting vectors. The starting point for the search of the lowest eigenvalue needs to satisfy only one constraint $x_{q1} \equiv \vec{X}_1^T \vec{e}_q = 1$ which is trivially satisfied by dividing through out by the qth component. The minimization algorithm using conjugate gradient method generate a sequence of vector which, in the limit, tend directionally to the minimum eigenvalue onthe search space. Thus the qth component of the vectors so generated have to be maintained as unity, so as to satisfy the constraint Eqn. $x_{q1} = \vec{X}_1^T \vec{e}_q = 1$ continuously in the space. This is achieved by setting

the qth component of the gradient vector at the particular point equal to zero. In other words, no 'move' is made in the qth direction of the search space. The starting point for the search of second eigenvalue has to satisfy an additional constraint Eqn. $\vec{X}_2^T [M] \vec{X}_1 = 0$, and this is achieved by Schmidt orthogonalization.

Let $\vec{X}^{(o)}$ be some initial estimate.

Therefore, $\vec{X}_2^{(o)} = \vec{X}_2^{(o)} - (\vec{X}_2^{(o)T} \vec{V}_1) \vec{V}_1$

satisfies the above constraint if $\vec{V}_1^T \vec{V}_1 = 1$.

Since $\vec{V}_1 \equiv [M] \vec{X}_1$, where $\vec{X}_1$ is the eigenvector corresponding to the first (lowest) eigenvalue.

The starting point for the search of the subsequent eigenvalues, is obtained by passing the initial estimate through a projection matrix, where the projection has for $[M_1]$, the following

$$\underset{(n \times l-1)}{[M_1]} \equiv [\vec{V}_1, \vec{V}_2, \ldots, \vec{V}_{l-1}]$$

3.4 (b) FUNCTION EVALUATION:

It is necessary to have an efficient routine for function evaluation, in order to avoid the time which would be otherwise wasted inside the minimization procedure.

Now, R $(\vec{X})$ can be written as

$$R(\vec{X}) = \frac{\sum_{i=1}^{r} \vec{Y}_i^T [k_i] \vec{Y}_i}{\sum_{i=1}^{r} \vec{Y}_i^T [m_i] \vec{Y}_i}$$

where r is the number of descrete elements, $[k_i]$ and $[m_i]$ are, respectively the stiffness and mass matrices of the ith element of the discretized structure and $\vec{Y}_i$ is the displacement vector of the ith element corresponding to the generalized displacement vector $\vec{X}$.

The size of element stiffness and mass matrices are relatively much smaller than the size of the assembled stiffness and mass matrices of large complex structure. The stiffness and mass matrices of various kinds of one diemensional structural elements are given in Appendix - II.

3.5 (c) <u>Gradient Evaluation:</u>

The minimization algorithm require the evaluation of gradient vector at each cycle of theiteration. The gradient vector, $\vec{g}$, of the Rayleigh quotient function is given earlier. For determining the lowest eigensolution, the component of the gradient vector corresponding to the normalizing component of the eigenvector is set equal

to zero, in order to continuously satisfy the constraint imposed due to the normalisation of the eigenvector. For the intermediate eigensolution, the projection matrix $[P_1]$ is used to project this gradient vector, $\vec{g}$, on to the proper subspace of search, M - orthogonal to the previously determined eigenvector. The component of the gradient vector, $\vec{g}_p$, corresponding to the normalizing component of the eigenvector turns out to be zero automatically, but a small number generally appears because of roundoff errors and this is removed by setting that component equal to zero.

3.6 (d) Evaluation of Step Length:

Once a direction of move $\vec{S}_i$ has been decided, we must determine α_i^* so that the. Function is minimized in that direction. Thus the problem of finding the step length is essentially the linear search problem which requires the determination of the α_i^* along $\vec{S}_i$ through $\vec{X}_i$ at which the value of the Rayleigh quotient function

$$R(\vec{X}_i + \alpha_i \vec{S}_i) = \frac{(\vec{X}_i + \alpha_i \vec{S}_i)^T [K] (\vec{X}_i + \alpha_i \vec{S}_i)}{(\vec{X}_i + \alpha_i \vec{S}_i)^T [M] (\vec{X}_i + \alpha_i \vec{S}_i)}$$

is a minimum, i.e.,

$$dR/d\alpha_i \Big|_{\alpha_i = \alpha_i^*} = R'(\alpha_i^*) = 0$$

In the general problem, no expression is available to determine α_i^*, so an interpolation approach is adopted. However, in the particular problem of the Rayleigh quotient, an explicit expression in α_i can be generated, which has the form

$$u\alpha_i^2 + v\alpha_i + w = 0$$

where
$$u = (\vec{S}_i^T [K] \vec{S}_0)(\vec{X}_i^T [M] \vec{S}_i) - (\vec{X}_i^T [K] \vec{S}_i)(\vec{S}_i^T [M] \vec{S}_i)$$

$$v = (S_i^T [K] S_i)(X_i^T [M] X_i) - (X_i^T [K] S_i)(S_i^T [M] S_i)$$

$$w = (X_i^T [K] S_i)(X_i^T [M] X_i) - (X_i^T [K] X_i)(X_i^T [M] S_i)$$

The two roots correspond to the maximal and minimal points of the Rayleigh quotient in the direction $\vec{S}_i$ through $\vec{X}_i$. The minimal function value corresponds to α_i^*.

CHAPTER - IV

APPLICATION - AND - RESULTS

In all the examples presented in this Chapter, the stiffness and mass matrices are taken from Appendix - II The distributed mass of the system was used to evaluate the components of the mass matrix (the 'consistent' mass matrix). IBM 7044 digital computer was used to obtain the numerical results and random vectors were taken as starting points for the minimization algorithm to obtain the eigensolution.

4.1 EXAMPLE (1):

As a simple application a slender solid cantilever rod shown in Fig. (1) is considered. The first three eigenvalues and eigenvectors of this simple structure were obtained by the successive minimization of the Rayleigh quotient in the appropriate subspace.

The cantilever rod shown in Fig.(1) is divided into three elements and each of them is modelled as a general planar beam element having six degrees of freedom. The cantilever Fig.(1) thus has a total of nine degrees of freedom. The first three eigenvalues and their associated mode shapes are given in Fig. 2(a) to 2(c).

The Table (1) compares the computed results with actual theoretical results obtained from closed form analytical expression.

4.2 Another illustrative example considered here is the portal frame shown in Fig. 3. Each member was modelled with a single beam element. Thus the structure has four degrees of freedom. A complete eigensolution was obtained by successive minimization of the Rayleigh quotient. The time taken was 24 seconds.

The computed frequencies are given in Table 2.

4.3 Finally a skew bridge structure with 12 degrees of freedom shown in Fig. 5, has been analyzed to obtain the first four eigenvalues and the associated eigenvectors by successive minimization of R.Q. The analysis is carried out for a grillage with two longitudinals, fixed at the extreme ends and their cross girders are assumed to be simply supported over them. The torsional rigidity of the members is neglected.

The computed values of frequencies obtained are compared with calculated values of the frequencies and the experimental values of frequencies as given in Ref. (11),

and are indicated in Table 3.

The first mode shape is plotted and is shown in Fig. 6.

The total time taken to compute the first four eigenvalues and the associated eigenvectors is four minutes approximately.

TABLE 1

Frequency No.	Actual Values in cs./sec.	Computed values in cs/sec.
1	5.16	5.099
2	31.599	31.799
3	88.38	88.99

TABLE 2

COMPUTED FREQUENCIES IN CS. /SEC. FOR PORTAL FRAME ARE:

1. 42.99
2. 241.25
3. 833.23
4. 1559.72

TABLE 3

Frequency No.	Calculated value of f. in cs/sec.	Experimental value of f in cs/sec	Computed value in cs/sec.
1	11.56	11.222	11.605
2	12.72	12.53	12.80
3	27.82	27.98	28.30
4	35.44	34.21	36.62

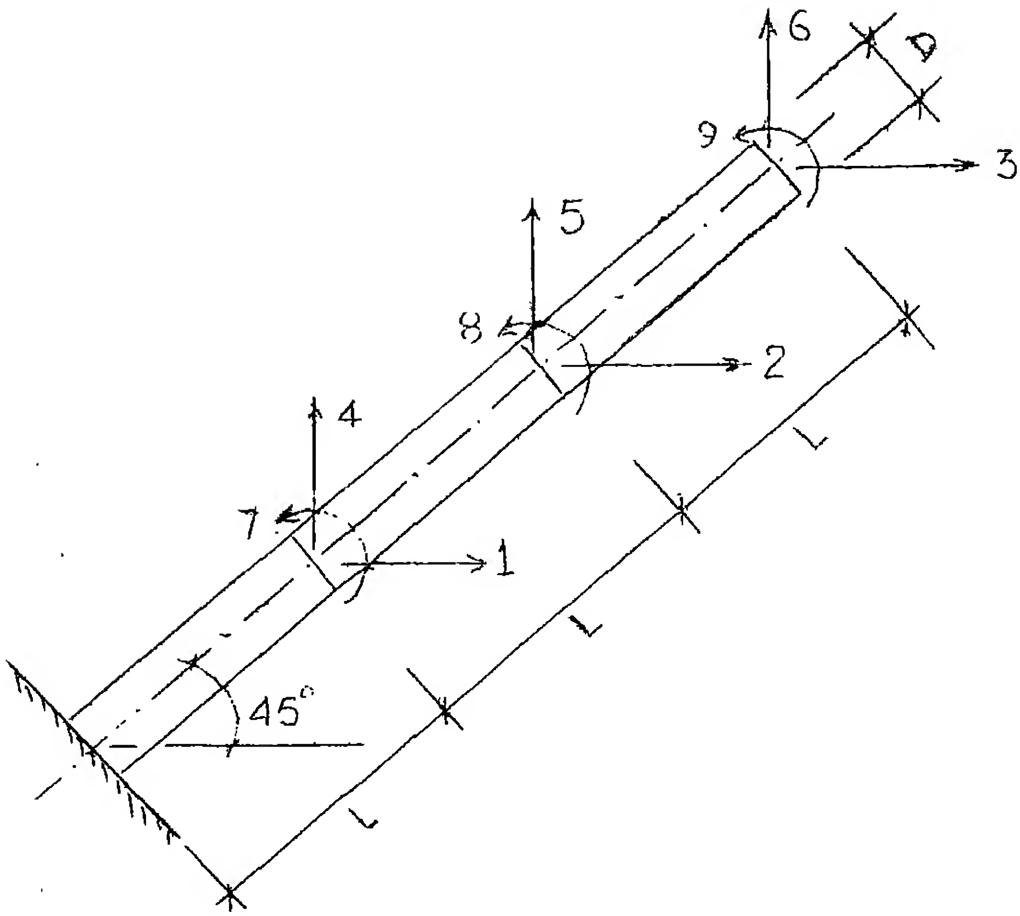


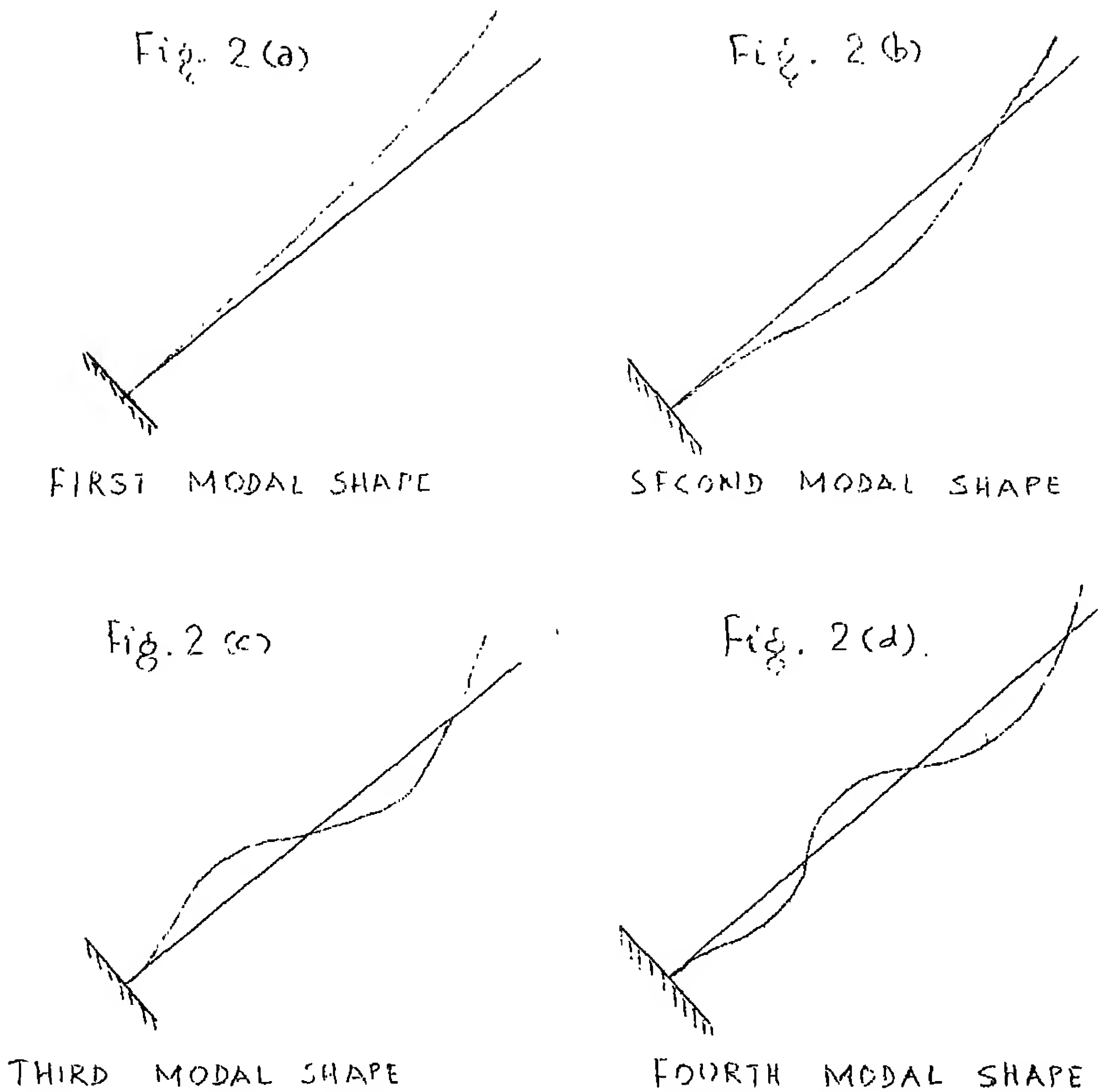
FIG. 1
SLENDER CANTILEVER ROD (9D.O.F.)

$L = 25.0$ in.

$D = 1.0$ in.

$E = 30.0 \times 10^6$ lbs./in^2

$\rho = 0.28$ lbs/in^2.



CANTILEVER MODE SHAPES

FIG. 2

(Refer Fig 1 - a slender Cantiler Rod inclined at 45°)

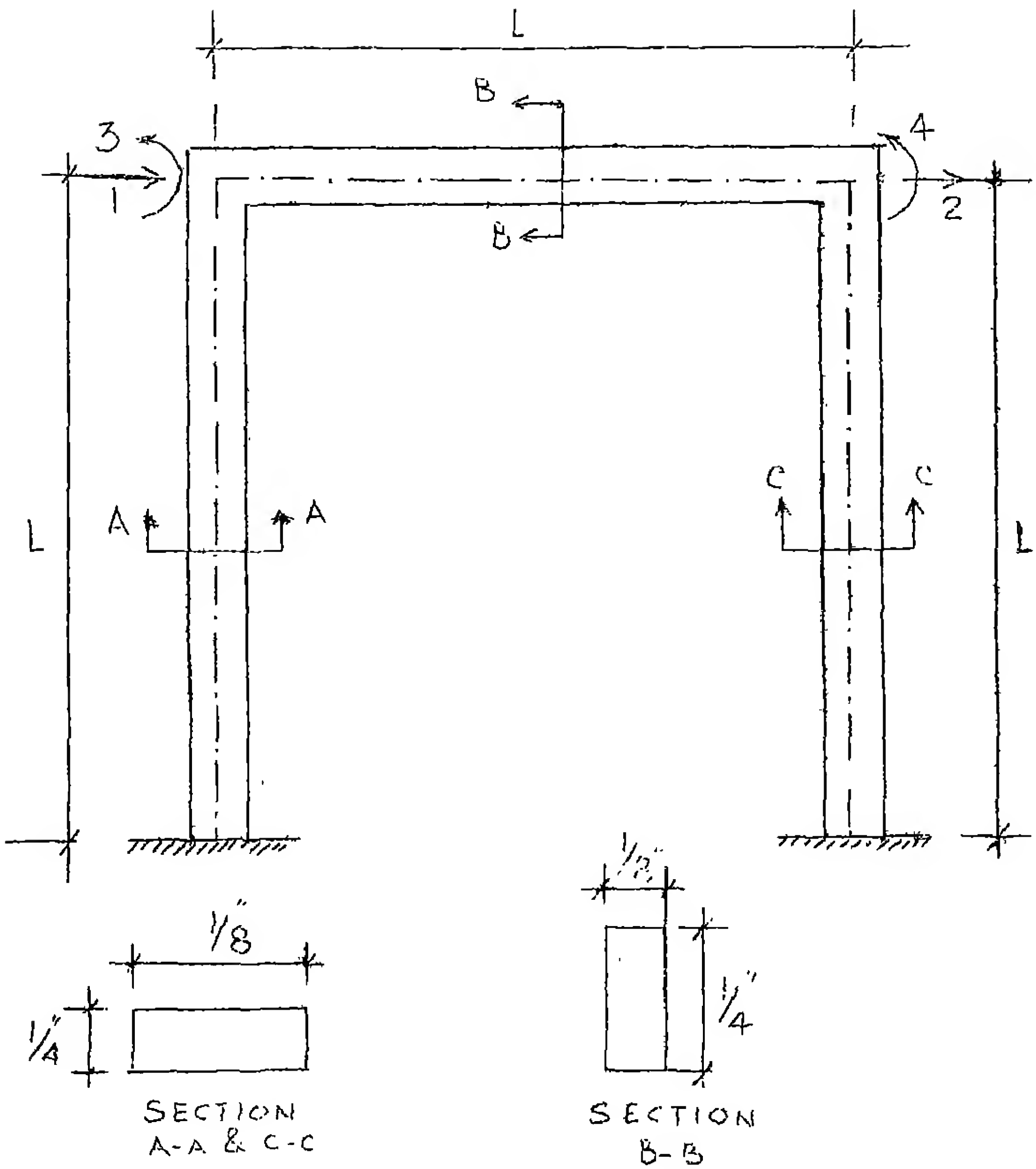


FIG. 3

PORTAL FRAME ; FOUR DEGREES OF FREEDOM SYSTEM.

$$L = 25.4 \text{ cm } (10 \text{ inches})$$

$$E = 2.1 \times 10^6 \text{ kgs/cm}^2 \; (30.0 \times 10^6 \text{ lbs/in}^2)$$

$$= 7.81 \text{ gm/cm}^3 \; (0.28 \text{ lbs/inch}^3)$$

DIFFERENT MODE SHAPES OF A PORTAL FRAME.

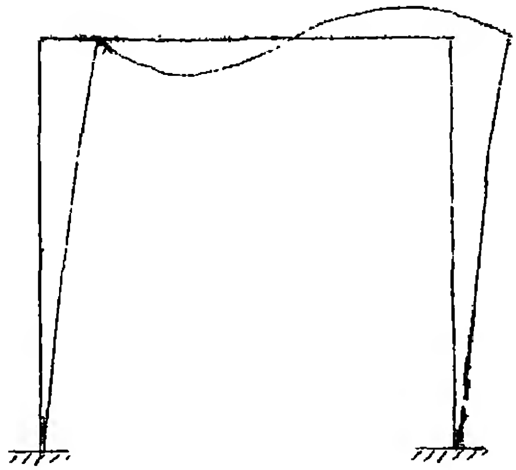


FIG. 4 (a)
FIRST MODAL SHAPE

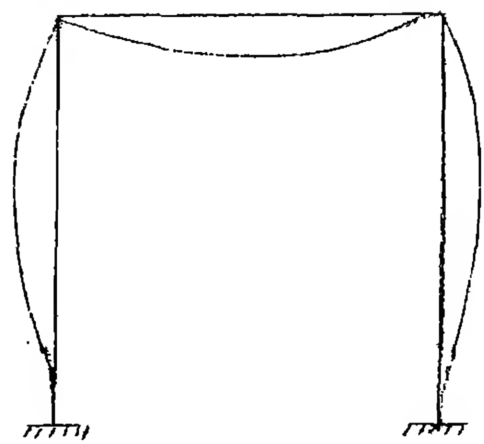


FIG. 4 (b)
SECOND MODAL SHAPE

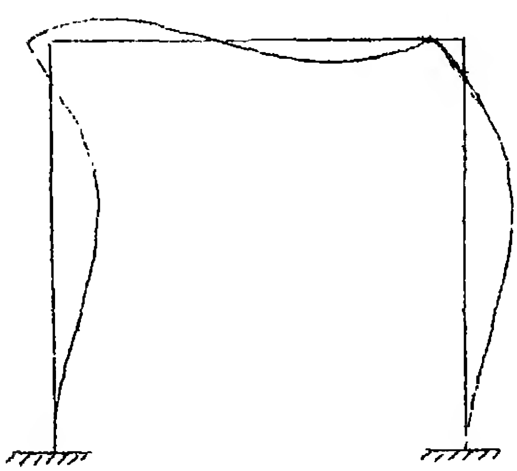


FIG. 4 (c)
THIRD MODAL SHAPE

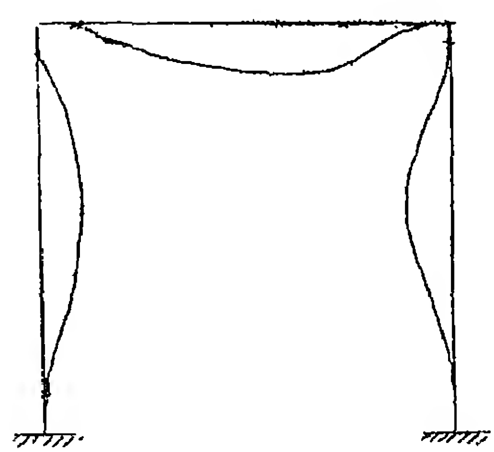
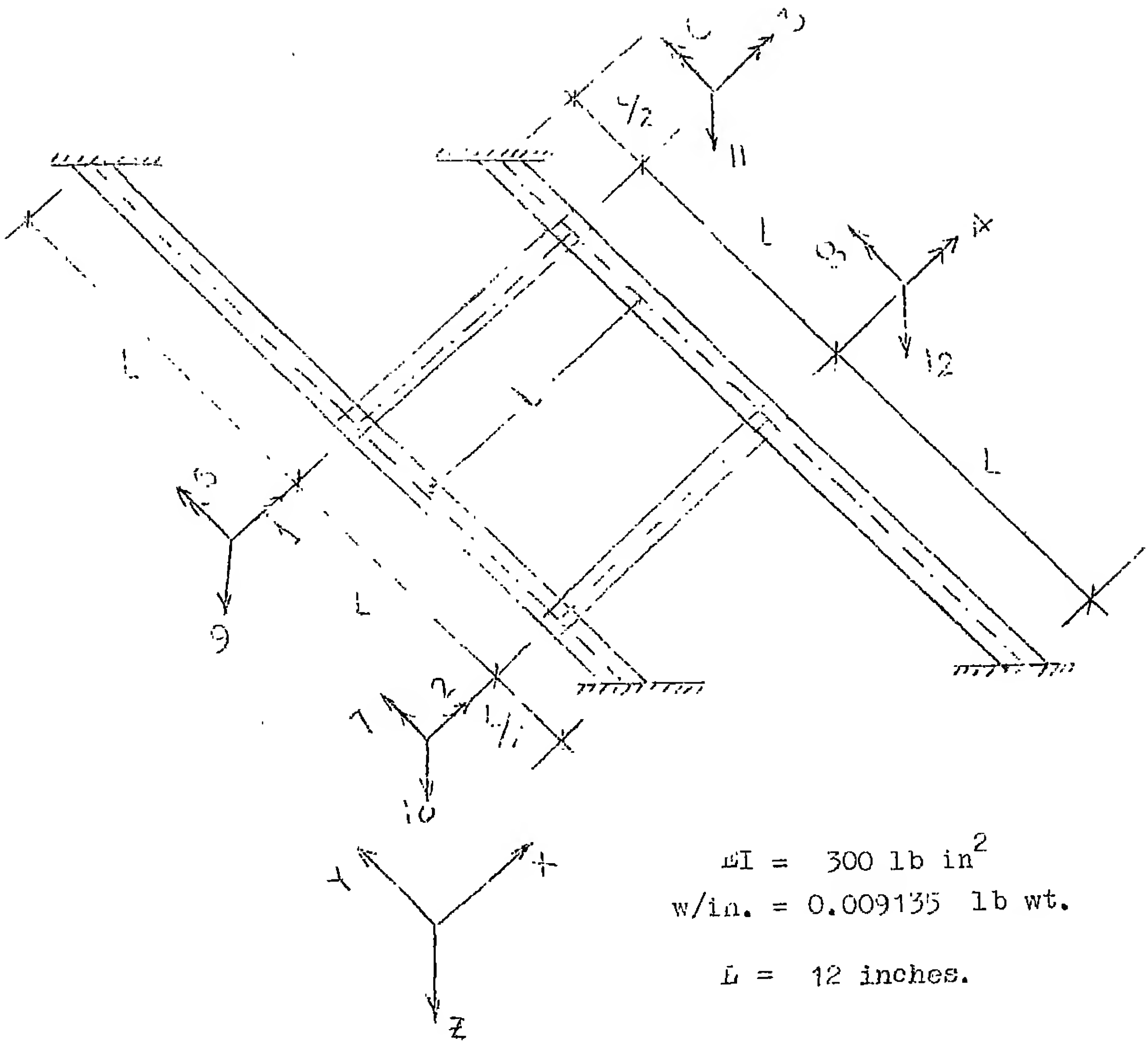


FIG. 4 (d)
FOURTH MODAL SHAPE.

FIG. 4



$EI = 300$ lb in^2

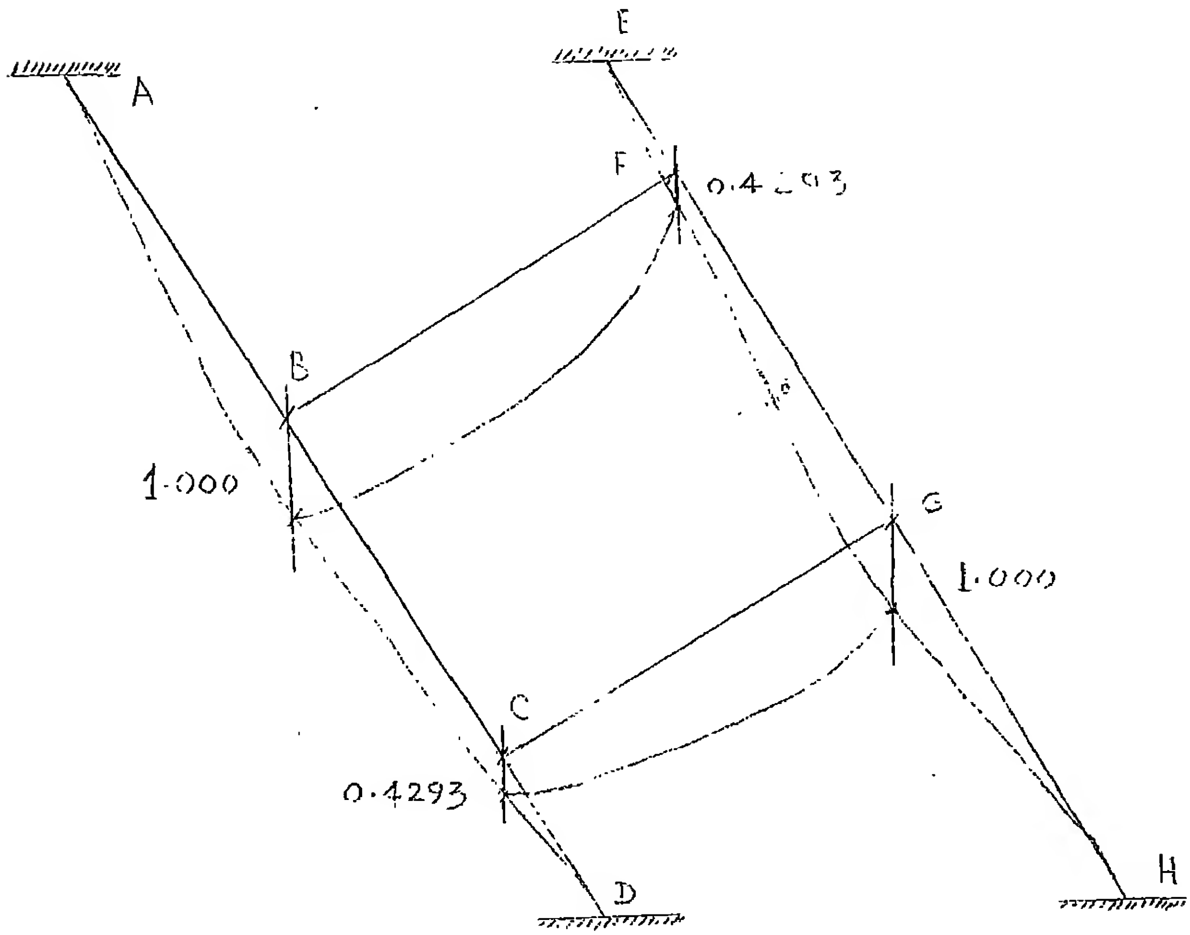
$w/in. = 0.009135$ lb wt.

$L = 12$ inches.

FIG. 5

SKEW BRIDGE; 12 DEGREES OF FREEDOM

MODAL SHAPE OF A SKEW BRIDGE



First Mode Shape of a Skew Bridge shown in Fig. 5.

FIG. 6

APPENDIX - I

LIST OF REFERENCES

1. Wilkinson, J.H., The Algebraic Eigenvalue Problem, (Oxford, Clarendon Press, 1965).

2. Anthony Ralston, A First Course in Numerical Analysis, (McGraw Hill Co.).

3. R.E.D. Bishop, Vibration (Cambridge University Press).

4. Fox, R.L. and Kapoor, M.P., A Minimization Method for the Solution of the Eigenproblem Arising in Structural Dynamics. AFFDL - TR - 68 - 150, Paper presented at the Wright Patterson Airforce Base Conference 1968.

5. J.S. Przemieniecki, Theory of Matrix Structural Analysis, (McGraw-Hill Co.), 1968.

6. Harold, C. Martin., Introduction to Matrix Methods of Structural Analysis.

7. Rubinstein, Moshe, F., Matrix Computer Analysis of Structures, (New Jersey: Prentice-Hall, 1966).

8. Gere, J.M. and Weaver, William, Jr., Analysis of Framed Structures (New Jersey: D. Van Nostrand Company, 1965).

9. Jacobsen, Lydik, S., and Ayre, Robert, S., Engineering Vibrations with Applications to Structures and Machinery. (New York: McGraw Hill Book Co. 1958).

10. Hurty, W.C., and Rubinstein Moshe, F., Dynamics of Structures, (New Jersey: Printice hall, 1964).

11. Armstrong, I.D., The Natural Frequencies of Grillages, (International Journal of Mechanical Sciences, 1968, Vol. 10, pp. 43-55).

12. Armstrong, I.D., Tables of Dynamic Stiffness and Carryover Factors, (Civil Engineering Department, Heriot-Watt University 1967).

13. D.J. Allman and D.M. Brotton, 'Systematic Relaxation Method for the Determination of the Normal Modes and the Natural Frequencies of Vibrating Systems', (International Journal of Mechanical Sciences, 1968, pp. 129-141).

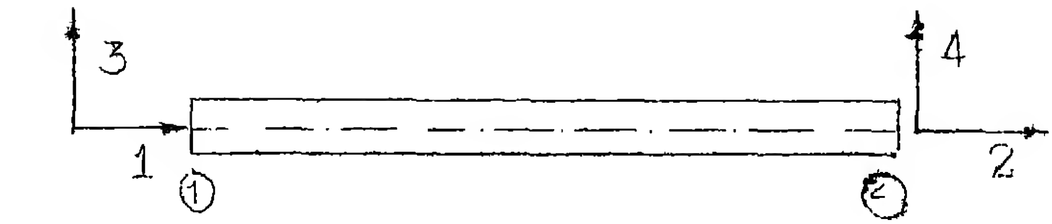
14. Mahesh, P. Kapoor, Automated Optimum Design of Structures Under Dynamic Response Restrictions, (A Thesis submitted to Case Western Reserve University in Partial fulfillment of the requirements for the Degree of Doctor of Philosphy, 1968).

15. Fox, R.L., Optimization Methods for Engineering Design, (Addison.Wesley 1971).

16. S.Z.H. Burney, Dynamics of Plane Structures by a Finite Element Method, (Symposium on Earthquake Engineering, Roorkee, Nov. 14-15-16, 1970).

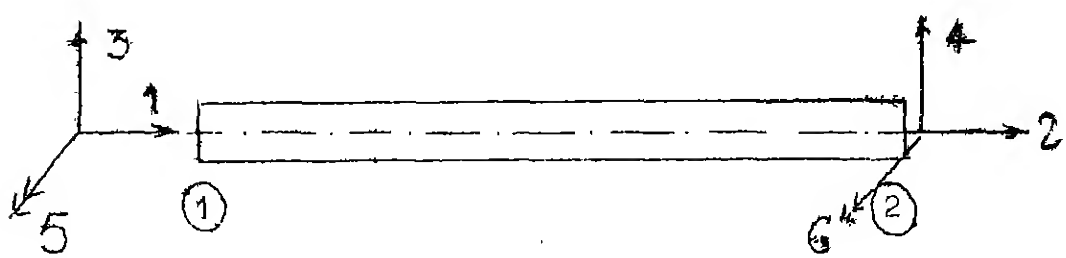
APPENDIX - II

STIFFNESS AND MASS MATRICES FOR VARIOUS TYPES OF STRUCTURAL ELEMENTS



A PLANAR TRUSS ELEMENT

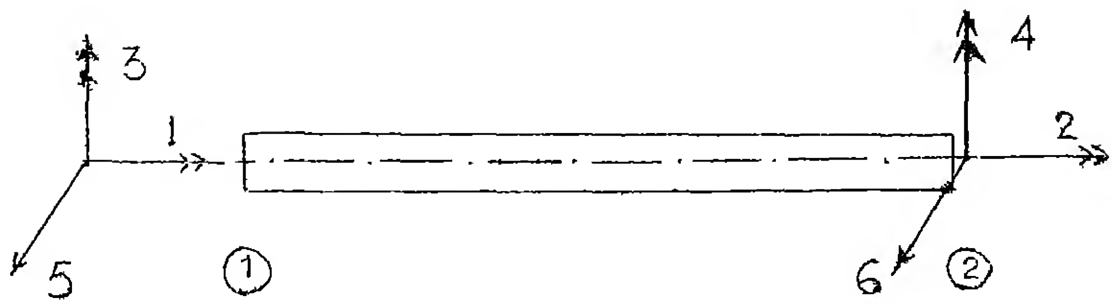
(4.D.O.F)



GENERAL BEAM ELEMENT AND A GRID ELEMENT CONSIDERING INPLANE VIBRATIONS

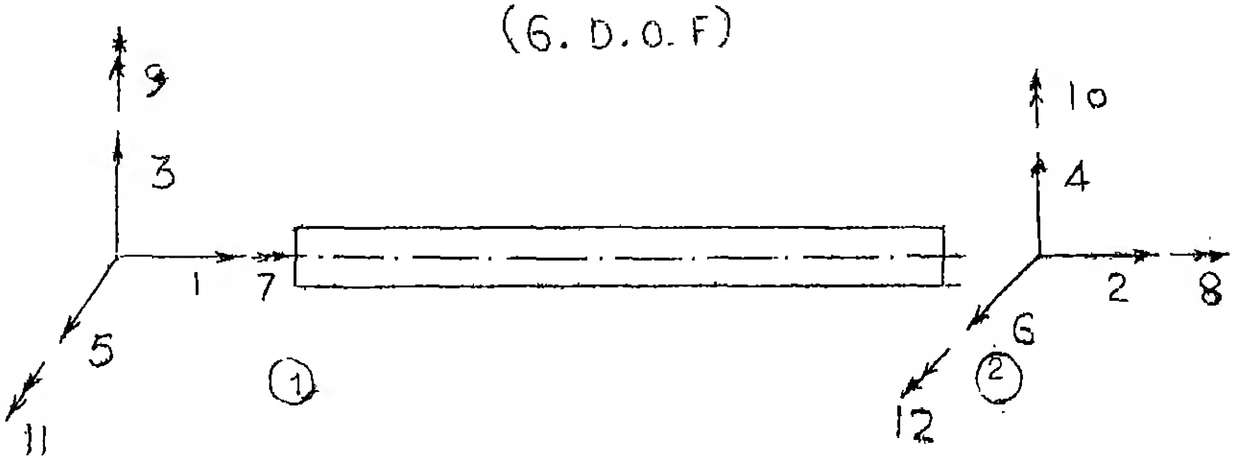
(6 D.O.F)

STIFFNESS AND MASS MATRICES FOR VARIOUS TYPES OF STRUCTURAL ELEMENTS



A GRID ELEMENT CONSIDERING LATERAL VIBRATIONS

(6. D.O.F)



A GENERAL SPACE FRAME ELEMENT

(12 D.O.F)

NOTE: DOUBLE ARROWS INDICATE ROTATIONS AND THE SINGLE ARROW INDICATES DISPLACEMENTS.

	1	2	3	4
1	EA_x/L	$-\dfrac{EA_x}{L}$		
2	$-\dfrac{EA_x}{L}$	$\dfrac{EA_x}{L}$		
3				
4				

STIFFNESS MATRIX FOR PLANAR TRUSS ELEMENT
(4 D.O.F)

$$\rho AL \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1/3 & 1/6 & & \\ 2 & 1/6 & 1/3 & & \\ 3 & & & & \\ 4 & & & & \end{array}$$

MASS MATRIX FOR PLANAR TRUSS ELEMENT (4 D.O.F.)

	1	2	3	4	5	6
1	$\frac{EA_x}{L}$	$-\frac{EA_x}{L}$				
2	$-\frac{EA_x}{L}$	$\frac{EA_x}{L}$				
3			$\frac{12\ EI_z}{L^3}$	$\frac{12\ EI_z}{L^3}$	$\frac{6EI_z}{L^2}$	$\frac{6EI_z}{L^2}$
4			$-\frac{12\ EI_z}{L^3}$	$\frac{12EI_z}{L^3}$	$-\frac{6EI_z}{L^2}$	$-\frac{6EI_z}{L^2}$
5			$\frac{6EI_z}{L^2}$	$-\frac{6EI_z}{L^2}$	$\frac{4EI_z}{L}$	$\frac{2EI_z}{L}$
6			$\frac{6EI_z}{L^2}$	$-\frac{6EI_z}{L^2}$	$\frac{2EI_z}{L}$	$\frac{AEI_z}{L}$

STIFFNESS MATRIX FOR GENERAL BEAM ELEMENT AND FOR GRID ELEMENT CONSIDERING INPLANE VIBRATIONS (6 D.O.F)

ρAL

	1	2	3	4	5	6
1	1/3	1/6				
2	1/6	1/3				
3			13/35	9/70	(11/210) L	$-\frac{13}{420}$ L
4			9/70	13/35	13/420 L	$-\frac{22}{420}$ L
5			$\frac{11}{210}$ L	$\frac{13}{420}$ L	$L^2/105$	$-\frac{3L^2}{420}$
6			$-\frac{13}{420}$ L	$-\frac{22}{420}$ L	$-\frac{3L^2}{420}$	$\frac{4}{420}$ L^2

MASS MATRIX FOR GENERAL BEAM ELEMENT AND FOR GRID ELEMENT CONSIDERING INPLANE VIBRATION (6 D.O.F)

1	2	3	4	5	6
GI_x/L	$-GI_x/L$				
$-\frac{GI_x}{L}$	$\frac{GI_x}{L}$				
		$4EI_y/L$	$2EI_y/L$	$-\frac{6EI_y}{L^2}$	$6EI_y/L^2$
		$2EI_y/L$	$4EI_y/L$	$-\frac{6EI_y}{L^2}$	$\frac{6EI_y}{L^2}$
		$-\frac{6EI_y}{l^2}$	$-\frac{6EI_y}{L^2}$	$\frac{12EI_y}{L^3}$	$-\frac{12EI_y}{L^3}$
		$\frac{6EI_y}{L^2}$	$\frac{6EI_y}{L^2}$	$-\frac{12EI_y}{L^3}$	$\frac{12EI_y}{L^3}$

STIFFNESS MATRIX FOR GRID ELEMENT CONSIDERING LATERAL VIBRATIONS (6 D.O.F)

ρAL

	1	2	3	4	5	6
1	$\frac{1}{3} I_x/A$	$I_x/6A$				
2	$I_x/6A$	$I_x/3A$				
3			$\frac{4}{420} L^2$	$-L^2/140$	$-\frac{11}{210} L$	$-\frac{13}{420} L$
4			$-L^2/140$	$\frac{4}{420} L^2$	$\frac{13}{420} L$	$\frac{22}{420} L$
5			$-\frac{11}{210} L$	$\frac{13}{420} L$	$13/35$	$9/70$
6			$-\frac{13}{420} L$	$\frac{22}{420} L$	$9/70$	$13/35$

MASS MATRIX FOR GRID ELEMENT CONSIDERING LATERAL VIBRATIONS

(6 D.O.F)

	1	2	3	4	5	6	7	8	9	10	11	12
1	$\frac{EA_x}{L}$	$-EA_x/L$										
2	$-\frac{EA_x}{L}$	EA_x/L										
3			$\frac{12EI_z}{L^3}$	$-\frac{12EI_z}{L^3}$							$\frac{6EI_z}{L^2}$	$\frac{6EI_z}{L^2}$
4			$-\frac{12EI_z}{L^3}$	$\frac{12EI_z}{L^3}$							$-\frac{6EI_z}{L^2}$	$\frac{6EI_z}{L^2}$
5					$\frac{12EI_y}{L^3}$	$-\frac{12EI_y}{L^3}$			$-\frac{6EI_y}{L^2}$	$-\frac{6EI_y}{L^2}$		
6					$-\frac{12EI_y}{L^3}$	$\frac{12EI_y}{L^3}$			$\frac{6EI_y}{L^2}$	$\frac{6EI_y}{L^2}$		
7							$\frac{GI_x}{L}$	$-\frac{GI_x}{L}$				
8							$-\frac{GI_x}{L}$	$\frac{GI_x}{L}$				
9					$-\frac{6EI_y}{L^2}$	$\frac{6EI_y}{L^2}$			$\frac{4EI_y}{L}$	$\frac{2EI_y}{L}$		
10					$-\frac{6EI_y}{L^2}$	$\frac{6EI_y}{L^2}$			$\frac{2EI_y}{L}$	$\frac{4EI_y}{L}$		
11			$\frac{6EI_z}{L^2}$	$-\frac{6EI_z}{L^2}$							$\frac{4EI_z}{L}$	$\frac{2EI_z}{L}$
12			$\frac{6EI_z}{L^2}$	$-\frac{6EI_z}{L^2}$							$\frac{2EI_z}{L}$	$\frac{4EI_z}{L}$

STIFFNESS MATRIX FOR SPACE FRAME ELEMENT (12 D.O.F.)

ρAL

	1	2	3	4	5	6	7	8	9	10	11	12
1	1/3	1/6										
2	1/6	1/3										
3			13/35	9/70							$\frac{11}{210}L$	$-\frac{13}{420}L$
4			9/70	13/35							$\frac{13}{420}L$	$-\frac{22}{420}L$
5					13/35	9/70			$-\frac{11}{210}L$	$\frac{13}{420}L$		
6					9/70	13/35			$-\frac{13}{420}L$	$\frac{22}{420}L$		
7							$\frac{1}{3}\ \frac{I_x}{A}$	$I_x/6A$				
8							$I_x/6A$	$I_x/3A$				
9					$-\frac{11}{210}L$	$-\frac{13}{420}L$			$\frac{4}{420}L^2$	$-L^2/140$		
10					$\frac{13}{420}L$	$\frac{22}{420}L$			$-\frac{L^2}{140}$	$\frac{4}{420}L^2$		
11			$\frac{11}{210}L$	$\frac{13}{420}L$							$\frac{L^2}{105}$	$-\frac{3L^2}{420}$
12			$-\frac{13}{420}L$	$-\frac{22}{420}L$							$-\frac{3L^2}{420}$	$\frac{4}{420}L^2$

MASS MATRIX FOR A SPACE FRAME ELEMENT (12 D.O.F.)

APPENDIX - III

COMPUTER PROGRAM

This appendix contains a description of each of the Fortran subprograms which makeup the Complete program for the free vibration analysis of structures composed of elements described in Appendix - II and the listing of each subprogram is appended at the end.

II-2 The Program:

The complete program for the free vibration analysis of structures consists of (A) a main program which reads the input data, evaluates the lengths of the various members of the structure, generates the starting random vector and prints the various input data. and (B) the other subroutines used are described below:

1 (a) Name of Subroutine: KRMR

(b) Argument List: M, LDF, RDF, NDFS, KL, ML, KR, MR, TR.

M : Total number of elements

LDF : Local degrees of freedom of an element

RDF : Number of degrees-of-freedom of an element in system co-ordinates.

NDFS : Number of degrees of freedom of system

KL : Element stiffness matrix in local-coordinate system

ML : Element mass matrix in local-coordinate system

KR : Element stiffness matrix in system co-ordinates

MR : Element mass matrix in system co-ordinates

TR : Transformation matrix for an element

(c) Common Variables: E, A, L, MIX, MIY, MIZ, RO, G, ALPHA, CX, CY, CZ, PE, QE, NPROB.

E : Modulus of Elasticity of the member

L : Length of the member

MIX : Moment of inertia of the member about x-axis

MIY : Moment of inertia of the member about y-axis

MIZ : Moment of inertia of the member about z-axis

RO : Weight per unit volume of the material of the member

G : *Modulus of rigidity of the member*

ALPHA : Angle of inclination of the member

CX, CY and CZ are respectively the X, Y and Z co-ordinates of the various nodes of the system respectively.

PE and QE : are respectively the left and right hand ends of a member.

NPROB : is a counter indicating the type of the problem. NPROB=1, indicates that the system consists of truss elements, 2 indicates that the system consists of either the general planar beam element or the grid element for inplane vibrations, 3 indicates that the system consists of grid elements for lateral vibrations and 4 indicates that the system consists of space frame elements.

2 (a) Name of Subroutine: FBMIN

(b) Argument List: M, RDF, NDFS, SR, KR, MR.

Most of them have been defined earlier.

SR : Assembly control table.

(c) Common Variables: SDIFF, AA, BB, NUMER, DENOM, VEC, BX, MIN, MINO, ITER, OMEGA, VALU, VECO, NVEC.

SDIFF : Scalar; Denotes convergence of Rayleigh quotient function; (MAIN)

AA : Vector; Denotes $[K]\vec{X}$, where $[K]$ is master stiffness matrix and $\vec{X}$ is generalized displacement vector.

BB : Vector; Denotes $[M]\vec{X}$, where $[M]$ is master mass matrix and $\vec{X}$ is generalized displacement vector.

NUMER : Scalar; Denotes $\vec{X}^T[K]\vec{X}$.

DENOM : Scalar; Denotes $\vec{X}^T [M] \vec{X}$.

VEC : Counter; Denotes the number of eigenvalues--eigenvectors being searched.

BX : Matrix ; the ith column of BX represents the vector $[M]\vec{X}_i$ where $[M]$ is master mass matrix and $\vec{X}_i$ represents the ith eigenvector.

MIN : Counter; Denotes the current component number of the eigenvector which is set to one.

MINO : Counter; Denotes the old component number of the eigenvector which was set to one.

ITER : Counter; Denotes the number of iteration of the Conjugate gradient method.

OMEGA : Vector; the components of OMEGA represent the frequencies (square root of eigenvalues).

VALU : Vector; the components of VALU represent the eigenvalues.

VECO : Matrix, the rows of VECO represent the eigenvectors

NVEC : Number of eigenvectors used for modal transformation.

(d) Purpose: To compute the eigenvalues and associated eigenvectors.

3. (a) Name of Subroutine: FANDG

(b) Argument List for this subroutine the argument list has already been defined.

Purpose: To compute the function (Rayleigh quotient) value and to evaluate the gradient vector to the Rayleigh quotient at any point.

4 (a) Name of Subroutine: PROD

(b) Argument List: Already defined.

Purpose: To compute the product

$$([N][N^T N]^{-1} [N]^T)\vec{W}$$

where $[N] \equiv B\vec{X}$

and $\vec{W}$ is any vector in system co-ordinates.

5(a) Name of Subroutine: INGEN

(b) Argument List: Already defined.

Purpose: To obtain $([N_l]^T [N_l])^{-1}$ from

$$([N_{l-1}]^T [N_{l-1}])^{-1}$$

6. (a) Name of Subroutine : KXMX

(b) Argument List: NDFS, SR, X, KR, MR, LBB

NDFS : Number of degrees of freedom of an element in system co-ordinates.

SR : Assembly control table

X : Vector in system co-ordinates

KR : Element stiffness matrix in system co-ordinates.

MR : Element mass matrix in system co-ordinates.

LBB : Logical; LBB =.TRUE. provides the normalized vector $BB \equiv [M]\vec{X}$ (Euclidean norm of the vector is equal to one).

(c) Common Variables: AA, BB, Y.

AA : Vector; Denotes $[K]\vec{X}$, where $[K]$ is master stiffness matrix and $\vec{X}$ is generalized displacement vector.

BB : Vector; Denotes $[M]\vec{X}$, where $[M]$ is master mass matrix and $\vec{X}$ is generalized displacement vector.

Y : Matrix; the ith row of the matrix denotes the displacement vector of the ith element corresponding to any generalized displacement vector $\vec{X}$ in system co-ordinates.

(d) Purpose: To compute the vectors $[K]\vec{X}$ and $[M]\vec{X}$ where $[K]$ and $[M]$ are respectively the master stiffness and mass matrices of the structure and $\vec{X}$ is any vector in system co-ordinates. Note that master stiffness and mass matrices are not assembled and the desired vectors are obtained by the use of only element stiffness and mass matrices.

7. (a) Name of Subroutine: TESTS
   (b) Argument List: Already defined.
   (c) Common Variables: NDFS, NVEC, X, BX, XBX , VEC.
   (d) Purpose: Tests for vector space, test for gradient, test for movevector.

A listing of each of the seven fortran subprograms which make up the complete program for the free vibration analysis of various types of structures for IBM 7044 digital computer follows in the subsequent pages.

```
BFTC MAIN
      INTEGER PE,QE,RDF,SR
      REAL L
      REAL  MIX,MIY,MIZ
      FORMAT(2X,10E11.6)
2     FORMAT(16F5.1)
      FORMAT(8(1X,E8.1))
      FORMAT(1X,25I3)
      FORMAT(//,3X, *NO. OF MEMBERS = *,I3,2X,* NO. OF NODES=*,I3, *LOCAL
     * D.O.F = *,I3,2X,* NO. OF DEGREES OF FREEDOM =*,I3,/0,2X,* NO. OF
     *EIGENVALUES AND EIGENVECTORS REQUIRED =*,I3,2X,* NPROB = *,I3//)
      FORMAT(1H0,2X,27H ASSEMBLY CONTROL MATRIX SR,//,2X,11H MEMBER NO.,
     15X,19H DEGREES OF FREEDOM,/,20(6X,I2,10X,6(I2,2X),/))
      FORMAT(1H0,2X,27H ASSEMBLY CONTROL MATRIX SR,//,2X,11H MEMBER NO.,
     *5X,19H DEGREES OF FREEDOM,/,20(6X,I2,10X, 4(I2,2X),/))
      FORMAT(1H0,2X,27H ASSEMBLY CONTROL MATRIX SR,//,2X,11H MEMBER NO.,
     *5X,19H DEGREES OF FREEDOM,/,130(6X,I2,10X,12(I2,2X),/))
6     FORMAT(13(1X,F5.1))
9      FORMAT(26F3.1)
8     FORMAT(16F5.1)
      DATA PI/3.1412/
      COMMON/GMAIN/VECO
      COMMON/KAUL/ KR,MP
      COMMON/HARI/IFF
      COMMON /SINGH/M,RDF,LDF
      COMMON/PATHAK/NVEC
      COMMON /LEVEL/ E,A,L,MIX,MIY,MIZ,RO,G,ALPHA,CX,CY,CZ,PE,QE,NPROB
      DIMENSION E(50),RO(50),CX(50),CY(50),CZ(50),PE(50),QE(50),SR(50,12KAU00280
     *),A(50),L(50),VECO(10,100),MIX(50),MIY(50),MIZ(50),G(50),ALPHA(50)KAU00290
      IFF=  0

      ----------- READ INPUT DATA -------------

      READ(5,5) M,N,LDF,RDF,NUFS,NVEC,NPROB
      READ NODAL CO-ORDINATES
      READ(5,98)  (CX(I),I=1,N)
      READ(5,98)   (CY(I),I=1,N)
      READ(5,98)    (CZ(I),I=1,N)
C     READ  SECTION CHARECTERISTICS
      READ(5,99)   (A(I),I=1,M)                                         KAU00350
      READ(5,2)  (ALPHA(I),I=1,M)
      READ(5,2)  (MIX(I),I=1,M)
      READ(5,2)  (MIY(I),I=1,M)
      READ(5,2)  (MIZ(I),I=1,M)
C     READ  MATERIAL  PROPERTIES
      READ(5,2)  (E(I),I=1,M)
       READ(5,3)  (G(I),I=1,M)
      READ(5,3)  (RO(I),I=1,M)
```

```
C
C      READ COINCIDENCE  ATRIX
       READ(5,5) (PE(I),I=1,M)
       READ(5,5) (QE(I),I=1,M)
C      READ ASSEMBLY CONTROL MATRIX
       READ(5,5) ((SP(I,J),J=1,RDF),I=1,M)
CCC     ECHO PRINT -----
       WRITE(6,6) ..M,LDF,NDFS,NVEC,..ROB
       WRITE(6,1)     (CX(I),CY(I),CZ(I),I=1,N)
       WRITE(6,1) (A(I),I=1,M)
       WRITE(6,1)  (MIX(I), IY(I),MIZ(I),I=1,M)
       WRITE(6,1)  (E(I),RO(I),I=1,M)
       WRITE(6,1)    (G(I),I=1,M)
       WRITE(6,5) (PE(I),QE(I),I=1, )
       WRITE(6,14)   (I,(SP(I,J),J=1,RDF),I=1,M)
CCC     GENERATE THE STARTING VECTORS----------------
       DO 12  I=1,NVEC
       DO 12   J=1,NDFS
12     VECO(I,J)=  RNDY1(J,U)
       DO 55   I=1,M
55     RO(I) = RO(I)/981.0
C      CALCULATE LENGTH OF MEMBERS AND DIRECTION COSINES
800    DO54I=1,M
       JT=PE(I)
       KT=QE(I)
54     L(I)=    SQRT((CX(KT)-CX(JT))**2+(CY(KT)-CY(JT))**2+,CZ(KT)-CZ(JT))**2)
      ***2)
       WRITE(6,316)  (L(I),I=1,M)
       CALL FR IN(..,RDF,NDFS,SP)
       STOP
       END


$IBFTC KFAR
        SUBROUTINE  KFAR(KR, ,1)
        INTEGER  RDF,PE,QE
        REAL L,L1,M1,.1,L2, 2,M2,L3,.3,N3,.D1,.D2,KR,MR,KL,ML
        REAL    MIX,MIY,MIZ
        COMMON /SINGH/ ,RDF,LDF
        COMMON /LEVEL/ E,A,L,MIX,MIY,MIZ,RO,G,ALPHA,CX,CY,CZ,PE,QE,NPROB
        DIMENSION E(50),A(50),L(50),MIX(50),MIY(50),MIZ(50),RO(50),G(50),    KAU
       *ALPHA(5.),CX(50),CY(50),CZ(50),KL(12,12),ML(12,12),KR(12,12),        KAU
       *MR(12,12),T.(12,12),T.(12,12),PE(50),QE(50)                          KAU
CC
  1     FORMAT(1X, 6E12.5)
 C      CALCULATE ELEMENT STIFFNESS AND MASS MATRICES
        DO 76 K=1, LDF
        DO 76 J=1, LDF
        KL(K,J)=   0.0
 76     ML(K,J)=   0.0
        KL(1,1)=   E(I)*A(I)/L(I)
        KL(1,2)=   -E(I)*A(I)/L(I)
```

```
KL(2,2)=   KL(1,1)
GO TO (10,20,30,40), NPROB
KL(3,3)=   (12.0*E(I)*MIZ(I))/(L(I)**3)
KL(3,4)=   -KL(3,3)
KL(4,4)=   KL(3,3)
KL(3,11)=   0.5*KL(3,3)*L(I)
KL(3,12)=   KL(3,11)
KL(4,11)=   -KL(3,11)
KL(4,12)=   KL(4,11)
KL(5,5)=   (12.0*E(I)*MIY(I))/(L(I)**3)
KL(5,6)=   -KL(5,5)
KL(5,9)=   0.5*KL(5,6)*L(I)
KL(5,10)=   KL(5,9)
KL(6,6)=   KL(5,5)
KL(6,9)=   0.5*KL(6,6)*L(I)
KL(6,10)=   KL(6,9)
KL(7,7)=   G(I)*MIX(I)/L(I)
KL(7,8)=   -KL(7,7)
KL(8,8)=   KL(7,7)
KL(9,9)=   4.0*E(I)*MIY(I)/L(I)
KL(9,10)=   0.5*KL(9,9)
KL(10,10)=   KL(9,9)
KL(11,11)=   4.0*E(I)*MIZ(I)/L(I)
KL(11,12)=   0.5*KL(11,11)
KL(12,12)=   KL(11,11)
GO TO 10
KL(3,3)=   (12.0*E(I)*MIZ(I))/(L(I)**3)
KL(3,4)=   -KL(3,3)
KL(3,5)=   0.5*KL(3,3)*L(I)
KL(3,6)=   KL(3,5)
KL(4,4)=   KL(3,3)
KL(4,5)=   -KL(3,5)
KL(4,6)=   -KL(3,5)
KL(5,5)=   4.0*E(I)*MIZ(I)/L(I)
KL(5,6)=   0.5*KL(5,5)
KL(6,6)=   KL(5,5)
GO TO 10
KL(1,1)=   G(I)*MIX(I)/L(I)
KL(1,2)=   -KL(1,1)
KL(2,2)=   KL(1,1)
MDI=   E(I)*A(I)/L(I)
KL(3,3)=   4.0*MDI
KL(3,4)=   2.0*MDI
KL(3,5)=   -6.0*MDI/L(I)
KL(3,6)=   6.0*MDI/L(I)
[illegible]   4.0*MDI
           KL(3,5)
           KL(3,6)
           /(L(I)**2)
```

```
L(6,6)=    KL(5,5)
...ATION OF THE MASS MATRIX--------------------------------
ONTINUE
D2= RO(I)*A(I)*L(I)
L(1,1)=    MD2*1.0/3.0
L(1,1)=    MD2*1.0/3.0
L(1,2)=    MD2*1.0/6.0
L(2,2)=    MD2*1. /3.0
D TO (110,120,13 ,14 ) ,    NI ROB
L(3,3)=    MD2*13.0/35.
L(3,4)=    MD2*9.0/70.0
L(3,11)=    MD2*L(I)*11.0/210.0
L(3,12)=    -MD2*L(I)*13.0/420.0
L(4,4)=    MD2*13.0/35.0
L(4,11)=    MD2*L(I)*13.0/420.0
L(4,12)=    -MD2*L(I)*22.0/420.0
L(5,5)=    MD2*13.0/35.0
L(5,6)=    MD2*9.0/70.0
L(5,9)=    -MD2*L(I)*11.0/210.0
L(5,10)=    MD2*L(I)*13.0/420.0
L(6,6)=    MD2*13.0/35.0
L(6,9)=    -MD2*L(I)*13.0/420.0
L(6,10)=    MD2*L(I)*22.0/420.0
L(7,7)=    (MD2*MIX(I))/(3.0*A(I))
L(7,8)=    0.5*ML(7,7)
L(8,8)=    ML(7,7)
(9,9)= 4.0*MD2*(L(I)**2)/420.0
(9,10)=    -MD2*(L(I)**2)/140.0
(10,10)=    ML(9,9)
(11,11)=    MD2*(L(I)**2)/1 5.0
(11,12)=    -MD2*3.0*(L(I)**2)/420.0
(12,12)=    ML(9,9)
 TO 110
(3,3)=    MD2*13.0/35.0
(3,4)=    MD2*9.0/70.0
(3,5)=    MD2*L(I)*11.0/210.0
(4,4)=    ML(3,3)
(3,6)=    -MD2*L(I)*13.0/420.0
(4,5)=    -ML(3,6)
(4,6)=    -ML(3,5)
(5,6)=    -MD2*(L(I)**2)*3.0/420.0
(5,5)=    MD2*(L(I)**2)/105.0
L(6,6)=    MD2*(L(I)**2)*4.0/420.0
 TO 110
(1,1)=    (MD2*MIX(I))/(3.0*A(I))
(1,2)=    0.5*ML(1,1)
(2,2)=    ML(1,1)
(3,3)=    MD2*(L(I)**2)*4.0/420.0
(3,4)=    -MD2*(L(I)**2)/140.0
(3,5)=    -MD2*L(I)*11.0/210.0
```

KAU01841

```
      ML(3,6)=    -MD2*L(I)*13.0/420.0
      ML(4,4) = ML(3,3)
      ML(4,5)=    -ML(3,6)
      ML(4,6)=    -ML(3,5)
      ML(5,5)=    MD2*13.0/35.0
      ML(5,6)=    MD2*9.0/70.0
      ML(6,6)=    MD2*13.0/35.0
110   CONTINUE
      DO 51 K=1,LDF
      DO 51 J=K,LDF
      KL(J,K)=    KL(K,J)
51    ML(J,K)=    ML(K,J)
CCC    ------ FORMATION OF TRANSFORMATION MATRIX ----------------------
      DO 81 K=1,LDF
      DO 81 J=1,LDF
81    TR(K,J)=    0.
      AA= ALPHA(I)
      JT=PE(I)
      KT=    QE(I)
      L1=    (CX(KT)-CX(JT))/L(I)
      M1=    (CY(KT)-CY(JT))/L(I)
      N1 = (CZ(KT) - CZ(JT))/L(I)
      IF(ABS(M1).GT.0.9999) GO TO 90
      DENOM=   (SQRT(L1**2+N1**2))
      L2=(-L1*M1*COS(AA)-N1*SIN(AA))/DENOM
      M2=    DENOM*COS(AA)
      N2=    (-M1*N1*COS(AA)+L1*SIN(AA))/DENOM
      L3= (L1*M1*SIN(AA)-N1*COS(AA))/DENOM
      M3=    DENOM*SIN(AA)
      N3=    (M1*N1*SIN(AA)+L1*COS(AA))/DENOM
      GO TO 91
90    L1=    0.0
      N1=0.0
      L2= -M1*COS(AA)
      M2=0.0
      N2= SIN(AA)
      L3= M1*SIN(AA)
      M3=0.0
      N3=    COS(AA)
91    TR(1,1)=    L1
      TR(2,2)=    L1
      TR(3,3)=    M2
      TR(4,4)=    M2
      TR(5,5)=    N3
      TR(6,6)=    N3
      GO TO (15,25,25,35)    ,NPROB
35    TR(1,3)=    M1
      TR(1,5)=    N1
      TR(2,4)=    N1
```

```
      TR(2,6)=   M1
      TR(3,1)=   L2
      TR(3,5)=   N2
      TR(4,2)=   L2
      TR(4,6)=   N2
      TR(5,1)=   L3
      TR(6,2)=   L3
      TR(5,3)=   L3
      TR(6,4)=   M3
      DO 92    K=1,6
      KK=K+6
      DO 92 J=1,6
      JJ=J+6
 92   TR(KK,JJ)=   TR(K,J)
      GO TO 80
 25   TR(1,3)=   M1
      TR(2,4)=   M1
      TR(3,1)=   L2
      TR(4,2)=   L2
      GO TO 80
 15   TR(1,1)=   L1
      TR(2,2)=   L1
      TR(3,3)=   L1
      TR(4,4)=   L1
      TR(1,3)=   M1
      TR(2,4)=   M1
      TR(3,1)=   -M1
      TR(4,2)=   -M1
 80   CONTINUE
C     TRANSFORM STIFFNESS MATRIX TO REFERENCE COORDINATE SYSTEM
      DO64J=1,LDF
      DO64K=1,RDF
      SUM=0.0
      DO65KK=1,LDF
 65   SUM=   SUM+KL(J,KK)*TR(KK,K)
 64   TS(J,K)=   SUM
      DO67J=1,RDF
      DO67K=1,RDF
      SUM=0.0
      DO68KK=1,LDF
 68   SUM=   SUM+TR(KK,J)*TS(KK,K)
 67   KR(J,K)=   SUM
C     TRANSFORM MASS MATRIX TO REFERENCE COORDINATE SYSTEM
 66   DO 70 J=1,LDF
      DO70K=1,RDF
      SUM=0.0
      DO71KK=1,LDF
 71   SUM=   SUM+ML(J,KK)*TR(KK,K)
 70   TS(J,K)=   SUM
 69   CONTINUE
 30
```

```
      DO 73 K = 1, RDF
      SUM=0.0
      DO74KK=1,LDF
74    SUM=   SUM+TR(KK,J)*TS(KK,K)
73    HR(J,K)=   SUM
72    CONTINUE
      RETURN
      END

$IBFTC FBMI.
      SUBROUTINE FBMIN( ,RDF,NOFS,SR)
C     THIS SUBROUTINE FINDS THE EIGEN VECTORS AND CORRESPONDING EIGEN
C    1-VALUES BY THE MINIMISATION OF THE RAYLEIGH QUOTIENT Z. THE
C    2CONJUGATE METHOD IS USED FOR MINIMISATION.
      INTEGER RDF,SR,VLC
      REAL NUMER,NUMER1,NUMER2,NUM,NP,  P,   NP ,KR,HR
      COMMON/PATHAK/NVEC
      COMMON/BBLOCK/AA,BB
      COMMON/CBLOCK/SUMER,DENOM
      COMMON/UBLOCK/OMEGA
      COMMON/JBLOCK/MI ,MI  ,ITER
      COMMON/DBLOCK/VEC
      COMMON/GBLOCK/BX
      COMMON/GMAIN/VECO
      COMMON/HARI/IFF
      DIMENSION SR(50,12),X(1  ),G(100),S(100),XEN(100),GXEN(100),AA(100KAU03
     *),BB(100),BX(100,10),XBX(1 0), (100), P(100),   P(100),XNR(100KAU03
     *,10),P(100),VALU(100),VECO(10,100),KR(12,12),HR(12,12),OMEGA(100 ,KAU03
     *TEMP(1  ),IIT(1  ),SX(1  ),CV(100)                                KAU03
1      FORMAT(1X,8E15.3)                                                HLK03
 13 FORMAT(1H0,23H RESULTS FROM ITERITION,I8/)
 14 FORMAT (1H0,15H XEN   GXEN   FEN/(2X,10E11.6))
 15 FORMAT (1H0,32H GENERALISED DISPLACEMENT VECTOR,/(2X,10E11.6))
 16 FORMAT (1H ,10X,19HI-1 FUNCTION VALUE,E20.9,5X,16HI FUNCTION VALUE
   1,2X,E20.8)
 17 FORMAT (1H0,2X,19H ORTHOGONALITY TEST,E19.8,5X,
   1          16H GRADIENT LENGTH,E19.8,5X,12H MOVE VECTOR,E19.8)
 18 FORMAT (1H0,12H MOVE VECTOR,/(2X,10E11.6))
 19 FORMAT (1H0,42H GENERALISED DISPLACEMENT VECTOR(STAQTING POINT),
   1        /(2X,10E11.6))
 20 FORMAT (1H1,2X,32H SEARCH  FOR  EIGENVECTOR  NO.  ,I2,2X,31H AND
   1CORRESPONDING  EIGENVALUE,//,)
 21 FORMAT (/,22H TEST FOR VECTOR SPACE)
 22 FORMAT (/,18H TEST FOR GRADIENT)
  3 FORMAT (/,21H TEST FOR MOVE VECTOR)
                        8H CHECK FOR EIGENVALUE NO. ,I2,/(2X,10E11.6))
                        2H ***** RESTARTED (****)
                        4H ***** STEEPEST DESCENT MOVE *****)
                        X,39H FRESH  SEARCH  FOR  EIGENVECTOR  NO.  ,I2,2X,31
                        SPONDING  EIGENVALUE,//,)
```

```
29    FORMAT (///, 1HO, 16H, EIGEN VALUES ARE,/, (2X, E 15.8))
30    FORMAT (///, 1HO, 16H FREQUENCIES ARE, /, (2X, E 15.8))
151   FORMAT(2X,10E11.4)
152   FORMAT(2X,20I5)
      DO 222 I=1,NDFS
      DO 222 J=1,NVEC
222   DX(I,J)=0.0
      DO 333 I=1,NVEC
      DO 333 J=1,NVEC
333   XNM(I,J)=0.0
      SDIFF=1.0E+08
      ITND=2*NDFS
 48   CONTINUE
      VEC=1
      WRITE(6,20) VEC
      DO 449 I=1,NDFS
  449 X(I)=VECO(VEC,I)
      XX=0.0
      MIN=0
      MINO=0
      DO 901 I=1,NDFS
      IF ((ABS(X(I))) .LE. XX) GO TO 901
      XX=X(I)
      MIN=I
      MINO=I
  901 CONTINUE
      DO 447 I=1,NDFS
447   X(I)=X(I)/XX
      WRITE (6,19) (X(I),I=1,NDFS)
453   IF(VEC.EQ. 1) GO TO 49
      CALL TESTS(NDFS,NVEC,NVECC,X,DX,XBX)
      WRITE (6,21)
      WRITE (6,1) (XBX(I),I=1,JKJ)
   49 ITER=0
350   CONTINUE
      IT=1
      NITER=10
      CALL FANDG(NDFS,SR,X,.TRUE.,F1,G)
      NUMER1=NUMER
      DENOM1=DENOM
      IF (VEC .EQ. 1) GO TO 85
      CALL TESTS (NDFS,NVEC,NVECC,G, X,XBX)
      WRITE (6,22)
      WRITE (6,1) (XBX(I),I=1,JKJ)
 4449 IF (VEC .NE. NDFS) GO TO 45
      VALU(VEC)=F1
      OMEGA(VEC)=SQRT(F1)
      DO 67 I=1,NDFS
   67 VECO(VEC,I)=X(I)
      WRITE (6,14) (X(I),I=1,NDFS),(G(I),I=1,NDFS),F1
      GO TO 6?
   85 DO 50 I=1,NDFS
   50 S(I)=-G(I)
      ITER=1
      WRITE (6,13) ITER
  200 CALL FANDG(NDFS,SR,S,.FALSE.,F,G)
```

```
      NUMER2=NUMER
      DENOM2=DENOM
      NUM=0.0
      DEN=0.0
      DO 51 I=1,NDFS
      NUM=NUM+X(I)*AA(I)
   51 DEN=DEN+X(I)*BB(I)
      U=(NUMER2*DEN)-(NUM*DENOM2)
      V=(NUMER2*DENOM1)-(NUMER1*DENOM2)
      W=(NUM*DENOM1)-(NUMER1*DEN)
      V=V/U
      W=W/U
      U=1.0
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 1200
      WRITE (6,1) U,V,W
 1200 TS1=-V/(2.0*U)
      TS2=((V**2-4.0*U*W)**0.5)/(2.0*U)
      TS3=-TS2
      ALPHA1=TS1+TS2
      ALPHA2=TS1+TS3
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 2200
      WRITE (6,1) ALPHA1,ALPHA2
 2200 IF ((ALPHA1 .LT. 0.0) .AND. (ALPHA2 .LT. 0.0)) GO TO 76
      IF ((ALPHA1.GT.0.0).AND.(ALPHA2.GT.0.0)) GO TO 60
      ALPHA=AMAX1(ALPHA1,ALPHA2)
      GO TO 70
   60 ALPHA=AMIN1(ALPHA1,ALPHA2)
   70 IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 3300
      WRITE(6,1) ALPHA
 3300    GO TO 71
   76 DO 79 I=1,NDFS
   79 XEM(I)=X(I)+ALPHA1*S(I)
      CALL FANDG (NDFS,SR,XEM,.FALSE.,F1,G)
      DO 80 I=1,NDFS
   80 XEM(I)=X(I)+ALPHA2*S(I)
      CALL FANDG (NDFS,SR,XEM,.FALSE.,F2,G)
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 4200
      WRITE (6,1) F1,F2
 4200 IF (F1 .LT. F2) GO TO 77
      ALPHA=ALPHA2
      GO TO 78
   77 ALPHA=ALPHA1
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 5200
   78 WRITE (6,1) ALPHA
 5200 DO 81 I=1,NDFS
   81 XEM(I)=X(I)+ALPHA*S(I)
      DO 82 I=1,NDFS
   82 X(I)=XEM(I)
             (6,25)
             STS(NDFS,NVEC,NVECC,X,BX,XBX)
             6,21)
             6,1) (XBX(I),I=1,JKJ)
             50

                  S
```

```
      TA=TA+G(I)*G(I)
   52 XEM(I)=X(I)+ALPHA*S(I)
      RMAX = 0.0
      IMAX=0
      DO 451 I=1,NDFS
      IF ((ABS(XEM(I))) .LE. RMAX) GO TO 451
      RMAX=ABS(XEM(I))
      IMAX=I
  451 CONTINUE
       IF(RMAX .LE. 10.0)  GO TO 454
      MINO=MIN
      MIN=IMAX
      DO 452 I=1,NDFS
  452 X(I)=XEM(I)
      XX=X(MIN)
      DO 450 I=1,NDFS
  450 X(I)=X(I)/XX
      WRITE (6,26)
      GO TO 453
  454 CALL FANDG(NDFS,SR,XEM,.TRUE.,FXEM,GXEM)
      NUMER1=NUMER
      DENOM1=DENOM
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 56
      IF(VEC.EQ.1) GO TO 86
      CALL TESTS (NDFS,NVEC,NVECC,XEM,BX,XBX)
      WRITE (6,21)
      WRITE (6,1) (XBX(I),I=1,JKJ)
 6665 CALL TESTS (NDFS,NVEC,NVECC,FXEM,BX,XBX)
      WRITE (6,22)
      WRITE (6,1) (XBX(I),I=1,JKJ)
   86 GS=0.0
      AG=0.0
      BE=0.0
      DO 53 I=1,NDFS
      GS=GS+S(I)*GXEM(I)
      AG=AG+S(I)*S(I)
   53 BE=BE+GXEM(I)*GXEM(I)
      BETA=BE/TA
      AG=SQRT(AG)
      BE=SQRT(BE)
      TEST=GS/(AG*BE)
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 6100
      WRITE (6,17) TEST,BE,AG
 6100 DO 54 I=1,NDFS
   54 S(I)=-GXEM(I)+BETA*S(I)
      IF(VEC.EQ.1) GO TO 6200
      IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 8200
      CALL TESTS (NDFS,NVEC,NVECC,S,BX,XBX)
      WRITE (6,23)
      WRITE (6,1) (XBX(I),I=1,JKJ)
 6200 IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 8200
 8200 WRITE(6,16) F1, FXEM
```

```
2 9 DIF=F1-FXEM
    IF (DIF .LT. 0.0) GO TO 400
 96 DEL=DIF*SDIFF
    DEB=DEL/10.0
    IF (DEL .LT. F1) GO TO 400
    IF ((DEB .LT. F1) .AND. (ABS(TEST) .LT. 0.1)) GO TO 400
 99 DO 55 I=1,NDFS
    G(I)=GXEM(I)
 55 X(I)=XEM(I)
    F1=FXEM
    ITER=ITER+1
    IF (ITER .GT. ITMD) GO TO 30
    IF(ITER.GT.NITER) IT=IT+1
     NITER=10*IT
    IF(ITER.GT.1.AND.ITER.LT.NITER) GO TO 9200
    WRITE (6,13) ITER
DC    GO TO 200
300 CONTINUE
    WRITE (6,14) (XEM(I),I=1,NDFS),(GXEM(I),I=1,NDFS),FXEM
    WRITE (6,27) VEC
    GO TO 250
400 CONTINUE
    WRITE (6,14) (XEM(I),I=1,NDFS),(GXEM(I),I=1,NDFS),FXEM
    VALU(VEC)=FXEM
    OMEGA(VEC)=SQRT(FXEM)
    DO 401 I=1,NDFS
401 VECO(VEC,I)=XEM(I)
    CALL KXMX(NDFS,SR,XEM,.FALSE.)
    DO 68 I=1,NDFS
 68 CV(I)=AA(I)/BB(I)
    WRITE(6,24) VEC,(CV(J),J=1,NDFS)
250 IF (VEC .EQ. NVEC) GO TO 69
    VEC=VEC+1
    WRITE (6,20) VEC
908 CONTINUE
    CALL KXMX(NDFS,SR,XEM,.TRUE.)
    JJ=VEC-1
    DO 58 I=1,NDFS
 58 BX(I,JJ)=BE(I)
    DO 40 I=1,NDFS
 40 P(I)=VECO(VEC,I)
 28 IF (VEC .GT. 2) GO TO 57
    DEN1=0.0
    DO 64 I=1,NDFS
 64 DEN1=DEN1+P(I)*BX(I,1)
    DO 65 I=1,NDFS
 65 X(I)=P(I)-DEN1*BX(I,1)
    GO TO 92
 57 IF (VEC .GT. 3) GO TO 229
    DEN1=0.0
    DO 89 I=1,NDFS
 89 DEN1=DEN1+BX(I,1)*BX(I,2)
    RAS=(1.0-(DEN1)**2)
```

```
      XNN(1,1)=1. /RAS
      XNN(1,2)=-DEM1/RAS
      XNN(2,1)=-DEM1/RAS
      XNN(2,2)=1.0/RAS
      GO TO 72
  229 VEC=VEC-1
      CALL INCEP(NDFS,UX,XNN,DB)
      VEC=VEC+1
   72 CALL PROD(NDFS,,VEC,P,BX,X.. , P,..  ..P, .  ..P)
      DO 61 I=1,NDFS
   61 X(I)=P(I)-   PP(I)
   92 XX=0.0
      MIN=0
      DO 902 I=1,NDFS
      IF ((ABS(X(I)))  .LE. XX) GO TO 902
      XX=X(I)
      MIN=I
  902 CONTINUE
      DO 62 I=1,NDFS
   62 X(I)=X(I)/XX
      WRITE (6,10)  (X(I),I=1,NDFS)
      JKJ=VEC-1
      CALL TESTS (NDFS,NVEC,NVECC,  UX,XBX)
      WRITE (6,21)
      WRITE (6,1) (XBX(I),I=1,JKJ)
 5100 GO TO 49
   49 WRITE (6,20)  (VALU(I),I=1,NVEC)
      WRITE (6,30)  (OMEGA(I),I=1,NVEC)
      RETURN
      END

C
$IBFTC FANDG
      SUBROUTINE FANDG(NDFS,SR,X,LFG,F,G)
      INTEGER SR,VEC,BDF
      REAL  NUMER,M,NN,  G,  NG,   G
      LOGICAL LFG
      COMMON /SINGH/M,REF,LDF
      COMMON /PATHAK/NVEC
      COMMON/BLOCK/AA,BB
      COMMON/CBLOCK/NUMER,DENOM
      COMMON/DBLOCK/VEC
      COMMON/CBLOCK/BX
      COMMON/JBLOCK/ MIN, TWO,ITER
      COMMON/HARI/IFF
      DIMENSION    X(100),G(100),AA(100),BB(100), (100,10),NN(100,10),   KAUG
     10,12),BX(100,10),N  NG(100),NG(100),   C(100)                       KAUG
     T(2X,8E15.8)
     T (8E15.8)
```

```
      DO 33 I=1,NDFS
   33 N(I,K)=BX(I,J)
   31 CONTINUE
      VEC=LL+1
      GO TO 11
   12 JJ=VEC-1
      DO 26 I=1,NDFS
 26   BB(I) = BX(I,JJ)
      CALL INGEN (NDFS,N,NN,BB)
      DO 27 I=1,NDFS
   27 N(I,VEC)=BX(I,JJ)
 11   CONTINUE
      CALL PROD(NDFS,NVEC,G,N,NN,NG,NNNG,NNNNG)
      DO 37 I=1,NDFS
   37 G(I)=G(I)-NNNNG(I)
   38 G(MIN)=0.0
   24 RETURN
      END
C
$IBFTC PROD
      SUBROUTINE PROD(NDFS,NVEC,G,N,NN,NG,NNNG,NNNNG)
C     THIS INTERNAL SUBROUTINE PERFORMS THE THREE MATRIX VECTOR MULTIPLI
C     CATIONS   (N(NTN)-1)NT)G
      REAL N,NN,NG,NNNG,NNNNG
      INTEGER VEC
      COMMON/DBLOCK/VEC
      DIMENSION G(100),N(100,10),NG(100),NNNG(100),NNNNG(100),NN(100,10)KAU06
      DO 31 I=1,VEC
      SUM=0.0
      DO 32 J=1,NDFS
   32 SUM=SUM+N(J,I)*G(J)
      NG(I)=SUM
   31 CONTINUE
      DO 33 I=1,VEC
      SUM=0.0
      DO 34 J=1,VEC
   34 SUM=SUM+NN(I,J)*NG(J)
      NNNG(I)=SUM
   33 CONTINUE
      DO 35 I=1,NDFS
      SUM=0.0
      DO 36 J=1,VEC
   36 SUM=SUM+N(I,J)*NNNG(J)
      NNNNG(I)=SUM
   35 CONTINUE
      RETURN
      END
```

```
C
$IBFTC KXMX
      SUBROUTINE KXMX(NDFS,SR,X,LBB)
C     THIS SUBROUTINE POSTMULTIPLIES  THE ELEMENT STIFFNESS AND MASS MAT
C     RICES BY THE ELEMENT DISPLACEMENT VECTOR,Y, AND THEN ASSEMBLES THE
C     (K)X AND ( )X VECTORS.
      INTEGER SR,RDF
      REAL KR,MR,L
      COMMON/SIMCH/M,RDF,LDF
C     M IS NUMBER OF ELEMENTS
C     RDF  IS NUMBER OF DEGREES OF FREEDOM IN REFERENCE COORDINATE SYSTEM.
C     ND   IS NUMBER OF DEGREES OF FREEDOM OF THE SYSTEM
      LOGICAL LBB
      COMMON/BBLOCK/AA,BB
      COMMON/HARI/IFF
      DIMENSION  KR(12,12), MR(12,12),SR(100,12),X(100),Y(100,12),TS1(100KAU
     *,12),TS2(100,12),AA(100),BB(100)                                  KAU
C     ASSIGN SYSTEM DEGREES OF FREEDOM TO ELEMENT DISPLACEMENT VECTOR
      IFF=IFF+1
      REWIND 3
      DO 10 I=1,M
      DO 11 J=1,RDF
      Y(I,J)=0.0
      JJ=SR(I,J)
      IF (JJ) 110,11,110
  110 Y(I,J)=X(JJ)
   11 CONTINUE
   10 CONTINUE
C     CALCULATE  (K)X  AND  ( )X
      DO 12 I=1,M
      IF(IFF.GT.1) GO TO 26
      CALL KMYR(KR,MR,I)
      WRITE(3) ((KR(J,K),K=1,LDF),J=1,LDF),((MR(J,K),K=1,LDF),J=1,LDF)
      GO TO 27
26    READ(3)  ((KR(J,K),K=1,LDF),J=1,LDF),((MR(J,K),K=1,LDF),J=1,LDF)
27    DO 13 J=1,RDF
      SUM1=  0.0
      SUM2=0.0
      DO 14 K=1,RDF
      SUM1=  SUM1+KR(J,K)*Y(I,K)
14     SUM2=  SUM2+MR(J,K)*Y(I,K)
      TS1(I,J)=   SUM1
13    TS2(I,J)=   SUM2
   12 CONTINUE
      DO 18 I=1,NDFS
      AA(I)=0.0
   18 BB(I)=0.0
      DO 19 K=1,M
      DO 20 KK=1,RDF
      I=SR(K,KK)
      IF (I) 21,20,21
      AA(I)=AA(I)+TS1(K,KK)
      BB(I)=BB(I)+TS2(K,KK)
      CONTINUE
      CONTINUE
      IF (.NOT. LBB) GO TO 22
```

```
 23 RAS1=RAS1+BB(I)*BB(I)
    DO 24 I=1,NDFS
 24 BB(I)=BB(I)/((RAS1)**0.5)
22    RETURN
      END

C
$IBFTC INGEN
      SUBROUTINE INGEN(NDFS,N,NN,X)
C     THIS INTERNAL SUBROUTINE GENERATES THE INVERSE OF A 9*9 MATRIX,GIV
C     NVEC IS NUMBER OF EIGENVALUES-EIGENVECTORS REQUIRED
      INTEGER VEC
      REAL NN,N,NG,NNNG,NNNNG
      COMMON/PATHAK/NVEC
      COMMON/DBLOCK/VEC
      DIMENSION PN(100),TM(100,10),A1(100,10),A2(100),X(100),NN(100,10),KAUG
     * N(100,10),NG(100),NNNG(100),NNNNG(100)                             KAUG
      CALL PROD(NDFS,NVEC,X,N,NN,NG,NNNG,NNNNG)
      DO 10 I=1,NDFS
 10   PN(I)=X(I)-NNNNG(I)
      AO=0.0
      DO 11 I=1,NDFS
 11   AO=AO+PN(I)*PN(I)
      AO=1.0/AO
      KKK=VEC-1
      DO 12 I=1,KKK
          DO 13 J=1,KKK
 13   TM(I,J)=NNNG(I)*NNNG(J)
 12   CONTINUE
      DO 14 I=1,KKK
      DO 15 J=1,KKK
 15   A1(I,J)=NN(I,J)+AO*TM(I,J)
 14   CONTINUE
      DO 16 I=1,KKK
 16   A2(I)=-AO*NNNG(I)
      DO 17 I=1,KKK
      DO 17 J=1,KKK
 17   NN(I,J)=A1(I,J)
      JJ=VEC
      DO 18 I=1,KKK
      NN(JJ,I)=A2(I)
 18   NN(I,JJ)=A2(I)
      NN(JJ,JJ)=AO
```

44

```
      RETURN
      END
C
$IBFTC TESTS
      SUBROUTINE TESTS(NDES,NVEC,MVECC,Y,BY,XBY)
      INTEGER VEC
      COMMON/DBLOCK/VEC
      DIMENSION  Y(100),BY(100,10),XBY(100)
      CON1=0.0
      DO10I=1,NDES
   10 CON1=CON1+Y(I)*Y(I)
      CON1=((CON1)**0.5)
      JJ=VEC-1
      DO11J=1,JJ
      SUM=0.0
      DO12I=1,NDES
   12 SUM=SUM+BY(I,J)*Y(I)
      XBY(J)=SUM
   11 CONTINUE
      DO13J=1,JJ
   13 XBY(J)=XBY(J)/CON1
      RETURN
      END
$ENTRY
26
```